MASTER EN SCIENCES INFORMATIQUES

Conception d'une base de données portant sur des paramètres biologiques et études statistiques

Fries, François-Xavier; Marchant, Vincent

Award date:
1989

Awarding institution:
Universite de Namur

CONCEPTION D'UNE BASE DE DONNEES PORTANT SUR DES PARAMETRES BIOLOGIQUES ET ETUDES STATISTIQUES.

Institut d'informatique

Année 1988-1989

Promoteur: M. Noirhomme

Mémoire présenté pour l'obtention du titre de Licencié et Maître en informatique par
François-xavier Fries et
Vincent Marchant.

REMERCIEMENTS

Nous voulons remercier ici les personnes qui nous ont apporté leur aide pour la bonne réalisation de notre travail de fin d'étude.

Tout spécialement, nous adressons tous nos remerciements à la firme CONTINENTAL PHARMA de Mont-Saint-Guibert qui nous a accueilli cette année. Et plus particulièrement, au département R.I.S. (Research Information System) et à ses membres , parmi lesquels comptent:

Monsieur J.P. BERNARD, responsable du département, qui a bien voulu mettre à notre disposition le matériel informatique nécessaire ainsi qu'un environnement de travail adéquat et une ambiance agréable.

Monsieur O. CLOSSON, responsable du projet, qui nous a aidé à nous familiariser avec le langage de quatrième utilisé. Nous le remercions pour sa disponibilité de tout instant.

Monsieur G. HEALEY, statisticien, qui nous a aidé à préciser les besoins des utilisateurs en statistique.

Enfin, nous remercions madame M. NOIRHOMME, d'avoir bien voulu être notre promoteur de mémoire pour un sujet auquel nous tenions et pour nous avoir conseillé plus spécialement dans le dòmaine des statistiques.

PLAN

CHAPITRE 1 : I N T R O D U C T I O N

1. INTRODUCTION

Ce mémoire a été fait dans le cadre d'une entreprise pharmaceutique ,CONTINENTAL PHARMA , située à Louvain-la-Neuve. Le choix de ce type d'environnement pour la réalisation d'un travail de fin d'étude provient d'une double motivation.

D'une part , nous désirions mettre à profit la formation que nous avions acquise en Biologie tout en nous concentrant sur les aspects informatiques.

D'autre part , il nous a paru intéressant de sortir d'un contexte purement universitaire et de nous confronter à un cadre de travail professionnel.
Nous espérions mettre en pratique nos connaissances pour développer une application informatique. Nous nous sommes alors tout naturellement tourné vers une entreprise.

Dans ce travail , nous donnerons d'abord des considérations théoriques nécessaires à la bonne compréhension du projet. Nous parlerons ainsi des langages de quatrième génération , du logiciel de programmation utilisé car celui-ci est un langage de quatrième génération. Ensuite, nous expliquerons certains concepts statistiques dont nous nous servirons ainsi que du logiciel statistique utilisé.

Après cela , nous détaillerons ce qui doit être réalisé et la manière dont nous l'avons fait.

Nous terminerons en donnant les perspectives d'évolution du logiciel ainsi qu'une discussion sur l'utilisation pratique d'un langage de quatrième génération.

CHAPITRE 2 : B U T D U T R A V A I L

2. BUT DU TRAVAIL.

Il consiste à élaborer une base de données contenant des valeurs de référence pour des mesures de constituants biologiques afin de pouvoir réaliser des traitements statistiques à partir de celle-ci.

Les valeurs ainsi stockées servent à l'interprétation des études de toxicologie. Le but de telles études consiste à essayer d'identifier les effets éventuels dus à l'administration de médicament que l' on désire commercialiser. Elles portent habituellement sur différents aspects :

- La Biologie clinique. Celle-ci fait intervenir tous les tests sur les constituants biologiques (dosages).On parlera des statistiques de biologie clinique pour se référer aux mesures réalisées ici.

- La toxicologie. Celle-ci comporte le relevé de toutes les autres informations (poids corporels,consommation alimentaire...) nécessaires à l'interprétation complète d'une étude de toxicologie. On parlera ici des statistiques de toxicologie.

Il est nécessaire de disposer de valeurs dites "de référence" car elles permettent de comparer les effets d'un médicament sur des animaux auxquels on a admistré celui-ci, par rapport à des animaux (témoins) ne l'ayant pas reçu et qui constituent des "groupes de référence".

De plus, les statistiques à offrir aux utilisateurs étant bien précises, il fallait se limiter à leur seule implémentation dans un premier temps.

Enfin, on peut souligner que l'objectif était de réaliser l'entièreté du développement de l'application afin qu'elle puisse être opérationnelle le plus rapidement possible et qu'elle puisse servir effectivement aux personnes responsables des études de toxicologie.

CHAPITRE 3 : C O N S I D E R A T I O N S T H E O R I Q U E S

3. CONSIDERATIONS THEORIQUES.

3.1. LES LANGAGES DE QUATRIEME GENERATION.

1° Qu'est-ce qu'un langage de quatrième génération.

Nous sommes en train de vivre une révolution dans le domaine des langages informatiques.

En effet, les langages existants depuis longtemps et devenus des standards tel l'assembleur, le PL 1 ou le cobol, ne satisfont plus toujours les demandes actuelles.
Il est de plus en plus important de pouvoir donner des instructions à un ordinateur le plus facilement et le plus rapidement possible. Il suffit pour s'en convaincre de voir l'évolution tant du matériel que des besoins informatiques dans tous les secteurs.

Le nombre de personnes devant utiliser un ordinateur ne cesse d'augmenter sans qu'il y ait pour autant une formation adéquate qui leur soit procurée. Il faut donc des langages puissants avec lesquels les utilisateurs, les analystes et les programmeurs puissent développer facilement leurs propres apllications et cela au prix d'un effort minimum.

C'est pour ces raisons que de gigantesques efforts de mise au point de nouveaux langages de programmation ont été entrepris depuis quelques années. Selon leur nature et leur fonctionnalité, on les regroupe dans la catégorie des langages dits de quatrième (L4G) ou de cinquième génération.

Pour comprendre la hiérarchie de langages et les types de catégories qui existent, voici les différentes générations distinguées par les professionnels (v.fig.1).

Le langage machine constitue la première génération de langage développé sur ordinateur.
Il peut être considéré comme un répertoire d'instructions effectivement possibles pour un processeur particulier.

Chaque instruction est constituée d'une suite de symboles binaires . Elle doit préciser les adresses physiques des données.

Ce langage est d'ailleurs le seul qui soit directement compris par le processeur !

La seconde génération est le langage d'assemblage symbolique ou assembleur.

Il a la même structure que le langage machine mais les suites de symboles binaires ont été remplacées par des symboles mnémoniques; ce qui donne au programme une forme plus claire.

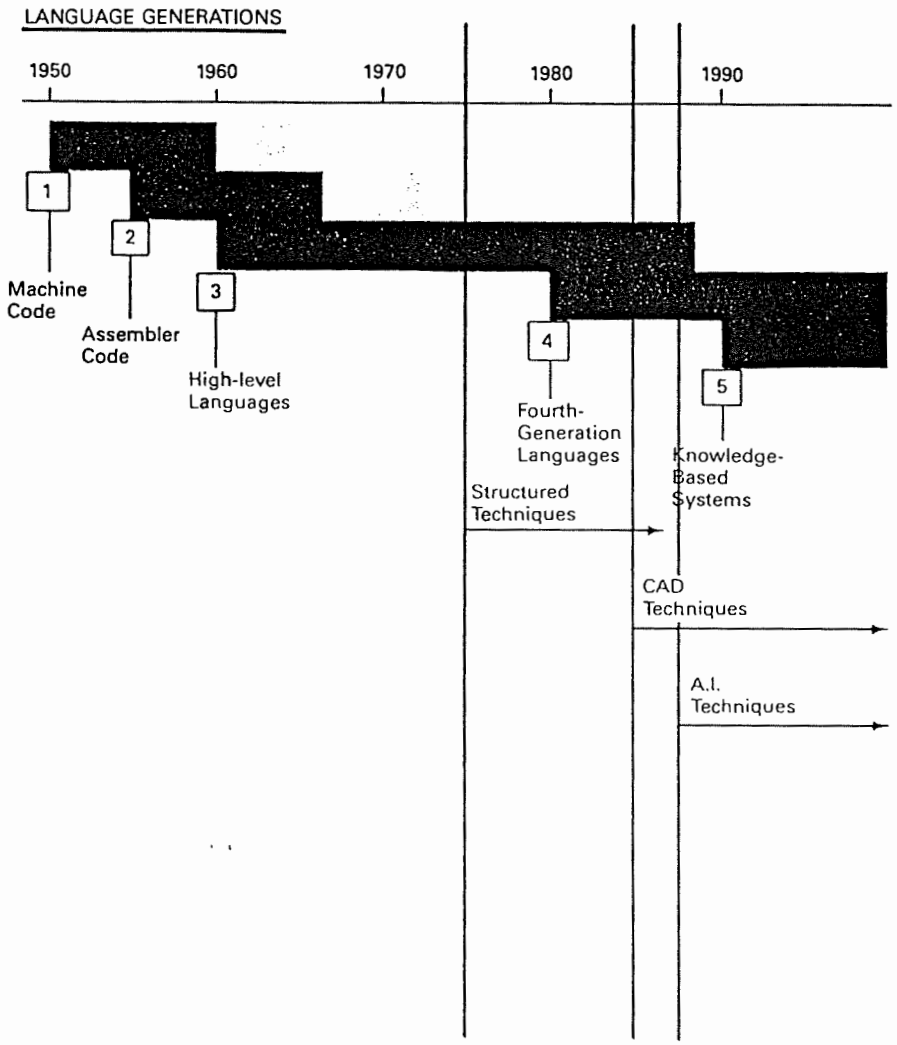


Figure 1 : Les différentes générations de langages.
(MARTIN 85 ,fig. 1-1)

Les problèmes rencontrés sont cependant le nombre d'instructions élémentaires important ainsi que le caractère non transportable des programmes réalisés sur un ordinateur particulier vers un autre ordinateur.

La troisième génération comprend les langages de haut niveau (Cobol,Basic,...etc.) qui se caractérisent par des instructions exprimant d'une manière plus compacte une opération nécessitant beaucoup d'instructions en Assembleur.
Il faut en outre transformer les programmes en langage machine par des interprêteurs ou des compilateurs.

Les problèmes qui y sont liés sont , entre autres ,le nombre de lignes de code relativement important par rapport aux L4G et la difficulté de modifier des systèmes complexes existants.

Les langages de quatrième génération ont été développés pour faciliter et améliorer la tâche de programmation.
Ils sont normalement plus faciles à utiliser, nécessitent un nombre d'instructions moindre, permettent un développement plus rapide des applications et entraînent une maintenance moins importante et plus aisée.

Ces langages de haut niveau sont dénommés "user friendly".
On les appelle parfois langages de base de données (mais certains n'en utilisent pas) ou langages non procéduraux (beaucoup d'entre eux contiennent cependant un code d'instructions).

La simplicité apparente des L4G permet donc de se concentrer sur ce que l'on veut faire, c'est-à-dire sur le but de l'application, plutôt que de se préoccuper de la réalisation pratique de celle-ci.
Il faut toutefois remarquer qu'il existe souvent un grand nombre de types de constructions différents dans ce genre de langage. Cela, contrairement aux langages de troisième génération pour lesquels la syntaxe peut varier fortement d'un langage à l'autre mais qui utilisent généralement le même ensemble de constructions.

Enfin, les langages de cinquième génération sont ceux touchant au domaine de l'intelligence artificielle.
Le plus connu d'entre eux est certainement Prolog. On peut également citer, en tant que langage privilégié pour l'élaboration de systèmes experts, le langage Lisp.

A la vue des diverses définitions qui ont été données à leur sujet, on peut constater qu'il est impossible de donner une définition exacte des langages de quatrième génération.

C'est ainsi que les constructeurs eux-mêmes, avides de ce type de dénommination pour leurs propres logiciels, n'hésitent pas à les proclamer de quatrième génération, sans toujours respecter les caractéristiques qui devraient leur être propres !

Cependant, si on devait définir un "bon" langage de quatrième génération, if faudrait qu'il ait plusieurs composants non-procéduraux pouvant être liés à des facilités procédurales (v.fig.2).

On y trouverait ,par exemple, outre le spécificateur des données, le générateur d'écran, le générateur de dialogue et le spécificateur de règles, un utilitaire procédural.

Pour mieux comprendre le phénomène quatrième génération, les points abordés dans ce chapitre expliqueront plus en détails les tenants et aboutissants de ces nouveaux langages de programmation.

2° Objectifs d'un langage de quatrième génération.

Les langages de quatrième génération représentent l'effort consenti il y a quelques années pour permettre aux informaticiens d'utiliser de manière plus efficiente les ordinateurs dont la puissance de traitement ne cesse d'augmenter.

Quelques principes sont à la base de la mise au point de ces langages [MARTIN 85]. Ainsi, parmi les objectifs essentiels, on peut relever :

- la facilité de développement d'application et la souplesse d'utilisation;
- la rapidité de formation à l'utilisation de base de tels langages;
- le gain de productivité de la part des programmeurs (rapidité de développement, nombre minimal d'erreurs);

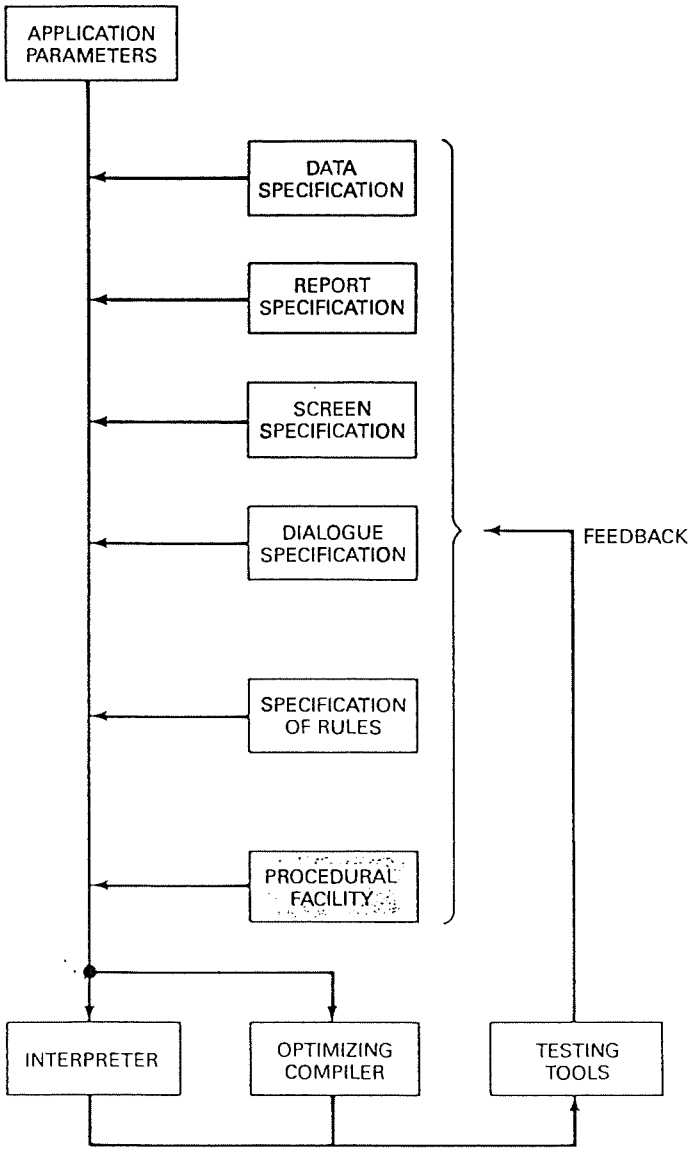


Figure 2 : Les composantes d'un langage de quatrième génération pour la conception d'applications (MARTIN 85, fig.2.5).

- la réduction des coûts et des efforts de maintenance;

- la standardisation des interfaces offertes : outils intégrés, dialogues et constructions syntaxiques faciles à retenir;

- la disposition d'un langage "user friendly".

Si les objectifs ont été bien définis pour la conception de ces langages, il n'en va pas de même quant à leur réalisation pratique. Beaucoup de langages offrent une partie seulement des facilités et atouts présentés ci-dessus.

Il faut donc rester prudent quant à l'analyse de leur spécificité pour que l'achat et l'utilisation des logiciels proposés se fasse en connaissance de cause et en fonction des besoins que l'on en aura.

3° Les catégories de langages.

Il existe de nombreux types de langages de programmation [MARTIN 85]. Les termes utilisés pour les décrire sont résumés ci-après.

Ne sont repris que ceux permettant de distinguer les différents langages de quatrième génération existants.

La plupart des L4G appartient à plus d'une de ces catégories de langages. Par exemple, ils peuvent utiliser à la fois du code procédural et non-procédural, ou bien, à la fois des procédures interactives d'entrée de données (via des écrans) et non interactives.

En ce qui concerne le mélange de code procédural et non procédural, il est d'ailleurs souhaitable d'en disposer dans un langage de quatrième génération. En effet, autant le premier étend la gamme des applications possibles, autant le code non procédural accélère et simplifie l'utilisation du langage.

A. Le langage PROCEDURAL.

Un langage procédural spécifie comment l'opération est effectuée. Le programmeur donne la liste d'instructions, détaillée et précise, pour chaque action à accomplir.

La majorité des langages de première, seconde et troisième génération sont de ce type.

B. Le langage NON PROCEDURAL.

Il spécifie ce qu'il faut faire sans décrire la séquence d'étapes à suivre (procédure) pour y arriver.

Il diffère du langage procédural par son aptitude à donner de meilleurs résultats sur la base de brèves expressions fournies par le programmeur.
Il permet d'aller plus vite en simplifiant l'utilisation du langage.

Par exemple, un générateur d'application dont l'utilisateur complète des formulaires pour décrire ce qu'il faut faire, n'est pas procédural.
L'utilisateur dit simplement ce qui doit être fait sans s'inquiéter de la procédure détaillée à suivre pour le réaliser.

C. Le langage DECLARATIF.

C'est une forme de langage non procédural.

Il spécifie un ensemble de faits et permet de poser des questions ou problèmes à résoudre qui utilisent ces faits.

Par exemple, la déclaration d'une équation du type pourcentage = X / Y * 100 , permet, en spécifiant n'importe quelle série de deux variables parmi les trois, de calculer la valeur de la troisième.

Il ne faut plus donner la séquence à suivre pour faire le calcul.

D. Le langage INTERACTIF ou de DIALOGUE.

C'est un langage avec lequel un programme est créé par dialogue interactif entre l'utilisateur et le logiciel.

Une grande diversité de formes de dialogue est utilisée par les langages de quatrième génération.

E. Le langage INTERACTIF PLEIN ECRAN.

Il permet au programmeur d'interagir avec un écran, de compléter des champs, de les changer de place, de les désigner ou bien encore de modifier des valeurs par défaut.

Ce type d'interaction via l'utilisation d'un écran tout entier est plus rapide et plus puissant que l'interaction orientée "on line" ; cela permet d'accélérer la conception de certains types de programmes.

Ici aussi, une grande variété de formes d'écran d'interaction existe dans les langages de quatrième génération.

F. Le langage de PROGRAMMATION GRAPHIQUE.

C'est un langage avec lequel on peut concevoir un programme interactivement avec des techniques graphiques.

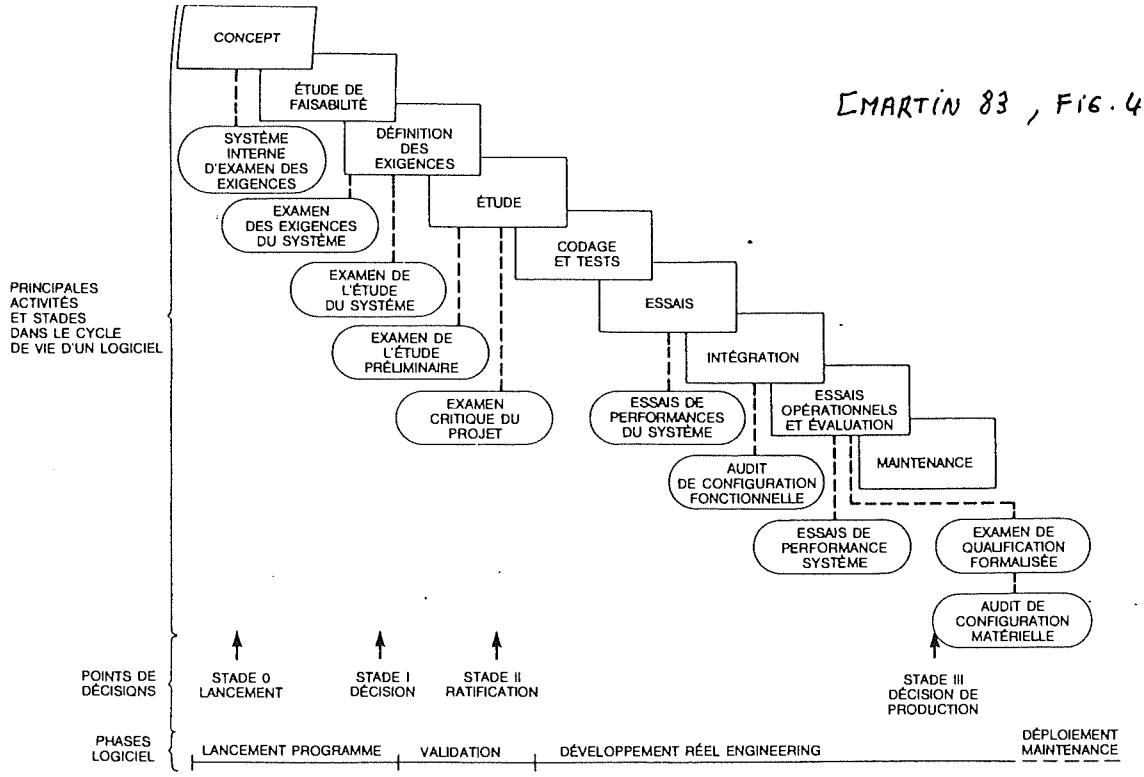
Des diagrammes peuvent être affichés à l'écran pour représenter des structures de contrôles, des arbres de décision ou pour générer du code.

Cette technique de programmation graphique peut être très rapide et puissante.

4° Impact sur le cycle de vie d'un projet informatique.

L'utilisation d'un langage de quatrième génération change le cycle de vie traditionnel de développement d'une application.

Ce cycle de vie traditionnel est le suivant :



[MARTIN 83, FIG. 4.1]

La gestion du développement d'une application informatique recquiert la création de certains documents et de résumés à chaque étape. Chaque phase du cycle de vie est bien expliquée, spécifiée et planifiée.

C'est important pour donner les grandes lignes directrices du projet, pour s'assurer que rien d'important n'a été oublié mais également pour pouvoir attribuer les tâches à chaque personne.

Ces techniques de gestion standard associées au cycle de vie traditionnel sont fortement implantées dans beaucoup d'organisations. Et pourtant, il y a encore de grands problèmes associés à ce cycle.

Actuellement, grâce à de nombreux outils et techniques qui n'existaient pas au moment de l'élaboration de ce cycle de vie standardisé et de sa reconnaissance dans les milieux informatiques, il faut modifier les différentes phases.

En effet, sont apparus :

- les langages non procéduraux
- les techniques de génération automatique de code
- les langages de spécifications
- les techniques de vérification rigoureuse
- les outils de conception graphique "on line"
- les outils de modélisation
- les langages de prototypage
- l'ingénierie de l'information

... etc.

Tous ces outils et techniques ont un impact majeur sur l'utilisation effective de l'ordinateur.

"Chaque étape du cycle de vie traditionnel doit être réexaminée pour maximiser le degré d'automatisation, pour assurer la flexibilité des systèmes vis-à-vis de changements ultérieurs ou encore afin de permettre aux utilisateurs eux-mêmes de manipuler les informations qu'ils possèdent" [MARTIN 83].

Différents types de cycle de vie sont apparus avec l'utilisation des langages de quatrième génération.

A. Le système "Ad hoc".

Les utilisateurs ou les analystes font des extractions de données, créent des données ou bien les manipulent, font des calculs, construisent des graphiques, ...etc.

Il n'y a aucune exigence d'analyse ou de spécifications écrites.

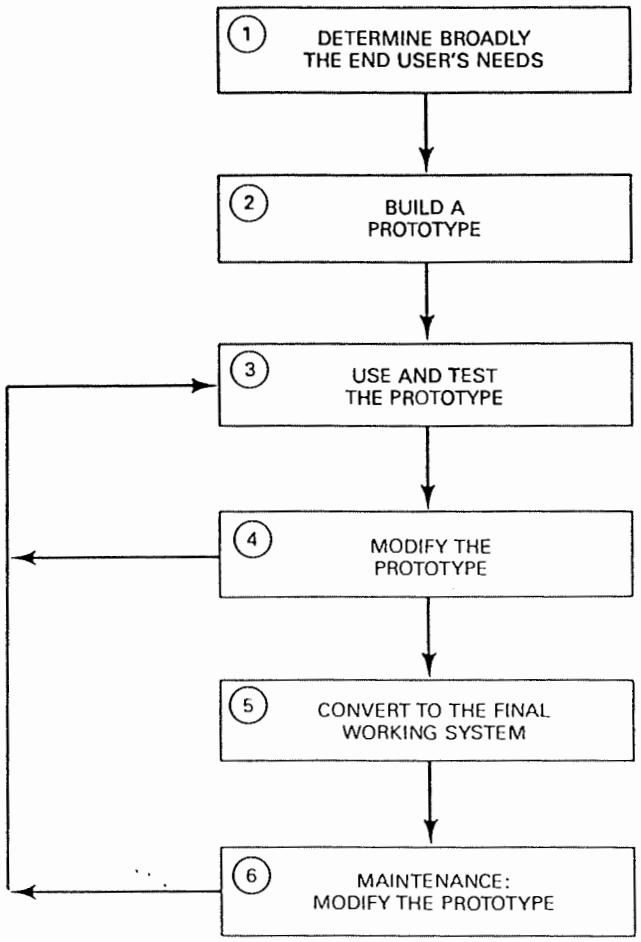


Figure 3 : Cycle de vie de type "Prototype"
(MARTIN 85, fig.16.1).

Si nécessaire, ce sont les créateurs de l'application eux-mêmes qui effectuent la maintenance. Il n'y a donc aucun cycle de vie formel.

La plupart des applications réalisées directement par les utilisateurs sont effectuées de cette manière.

B. Le cycle "Prototype".

L'analyste établit un prototype qu'il soumet directement à l'utilisateur. C'est d'après les remarques qui lui sont faites qu'il modifie le prototype.

De nombreuses versions peuvent être réalisées jusqu'à satisfaction de l'utilisateur (v.fig.3).

Par la suite, ce prototype devient le système de travail et l'analyste peut compléter sa documentation (de préférence réalisée "on line") et aider à l'utilisation du logiciel.

C. Le cycle "Prototype avec langage de troisième génération".

Comme précédemment, on effectue d'abord un prototype. Ensuite, au lieu d'en faire directement l'application finale, on le reprogramme avec un langage de troisième génération. Ceci permet d'optimiser les performances machines.

Le désavantage d'une telle pratique réside dans le fait que la capacité de faire des modifications rapidement, offerte par un langage de quatrième génération, est perdue !

Il est donc préférable de disposer d'un compilateur optimisant avec un L4G ou bien de disposer d'une base de données ou encore d'un système de fichiers conçu de manière performante au point de vue machine.

D. Le cycle "Spécification et langage de quatrième génération".

Tout ce qui concerne l'analyse et les spécifications est réalisé exactement de la même façon que dans le cycle de vie traditionnel. Mais le code exécutable est réalisé avec un générateur de code.

Cette technique est habituellement utilisée pour des systèmes complexes où il faut faire très attention aux spécifications.

Pour des raisons d'efficacité, il convient d'utiliser le L4G uniquement pour une partie du système ou bien encore de se servir de deux types de L4G ; l'un produisant un code optimisé et l'autre procurant une bonne flexibilité dans la génération d'états de sortie.
Cette méthode a été utilisée pour créer le système de manufacturation de XEROX [MARTIN 85].

E. Le cycle "Langage de spécification".

Le désavantage du cycle précédemment décrit réside dans le fait que les spécifications manuelles sont souvent inconsistantes, ambiguës, imcomplètes ou sujettes à mauvaise interprétation.

Ceci peut être évité si on utilise un langage de spécification. Celui-ci force à être très rigoureux au niveau des spécifications utilisées.

Ce type de langage peut être associé à un générateur de code.

A côté de ces différents types de cycle de vie pouvant exister lors de l'utilisation d'un langage de quatrième génération, il faut encore parler de la modélisation des données elles-mêmes.

Dans un environnement de quatrième génération, il est très avantageux de disposer d'une administration des données.

L'administrateur de données maintient un dictionnaire et un modèle des données, donnant une

	Développement classique d'applications	Générateur d'applications utilisé comme aide prototype suivie par la programmation	Développement d'applications sans programmeurs professionnels
Analyse des exigences.	Opération procédurale qui prend du temps et est souvent retardée par le grand retard des applications.	L'imagination de l'utilisateur est stimulée. Il peut travailler sur un écran avec un analyste pour développer les exigences.	L'imagination de l'utilisateur est stimulée. Il peut développer ses propres exigences ou travailler avec un analyste.
Document de spécifications.	Document très long. Ennuyeux. Souvent inadéquat.	Produit par aide de prototype. Précis et testé.	Disparaît.
Paraphe utilisateur.	L'utilisateur n'est souvent pas sûr qu'il paraphe. Il ne peut percevoir toutes les subtilités.	L'utilisateur voit les résultats et peut les modifier avant de parapher.	Pas de paraphe formel. Ajustement et modifications se font tout seuls.
Codage et essais.	Lent. Coûteux. Souvent retardé à la suite du retard de l'informatique.	Le prototype est transformé en code plus efficace. Relativement rapide et exempt d'erreurs.	Rapide. Peu coûteux. Disparaît dans une grande proportion.
Documentation.	Ennuyeuse. Prend beaucoup de temps.	Peut être partiellement automatisé. Formation interactive et réponse HELP créées en ligne.	Largement automatique. Formation interactive et réponses HELP sont créées en ligne.
Maintenance.	Lente. Coûteuse. Souvent en retard.	Souvent lente. Souvent coûteuse. Souvent en retard.	Processus continu avec utilisateur et analyste faisant des ajustements. Ces derniers peuvent généralement prendre quelques heures au lieu de quelques mois.

Figure 4 : Caractéristiques de trois types de développement d'applications.
(MARTIN 83, tableau 4.2)

description des données utilisées dans l'organisation. Un sous-ensemble de celles-ci est extrait en fonction des projets individuels.

Cela accroît considérablement la vitesse de développement, permet une communication précise des données entre développeurs mais cela assure également l'échange de données entre différents systèmes.

En conclusion, il est clair qu'il n'est plus possible de se baser sur un cycle de vie de développement tout à fait formel. Il y a différents types de cycle de vie possibles pour différents types de méthodologies et d'outils.

Au départ de tout projet important, il est essentiel de mettre au point un cycle de vie qui soit adapté et approprié aux méthodologies et aux outils dont on dispose à ce moment là.

Enfin, on peut considérer qu'il existe trois types de développement d'application [MARTIN 85]:

- le développement traditionnel avec un cycle de vie traditionnel.
- l'utilisation d'un langage de quatrième génération en tant qu'outil permettant la réalisation d'un prototype (cfr.point 5°).
- l'utilisation d'un langage de quatrième génération pour développer l'application toute entière. Le prototype éventuel devient l'application elle-même. Cette manière de faire peut être utilisée pour la plupart des traitements de données à caractère commercial.

La figure 4 reprend la liste des caractéristiques propres aux trois approches en ce qui concerne le développement d'une application.

Il faut noter que dans certaines grandes organisations les trois types peuvent exister.

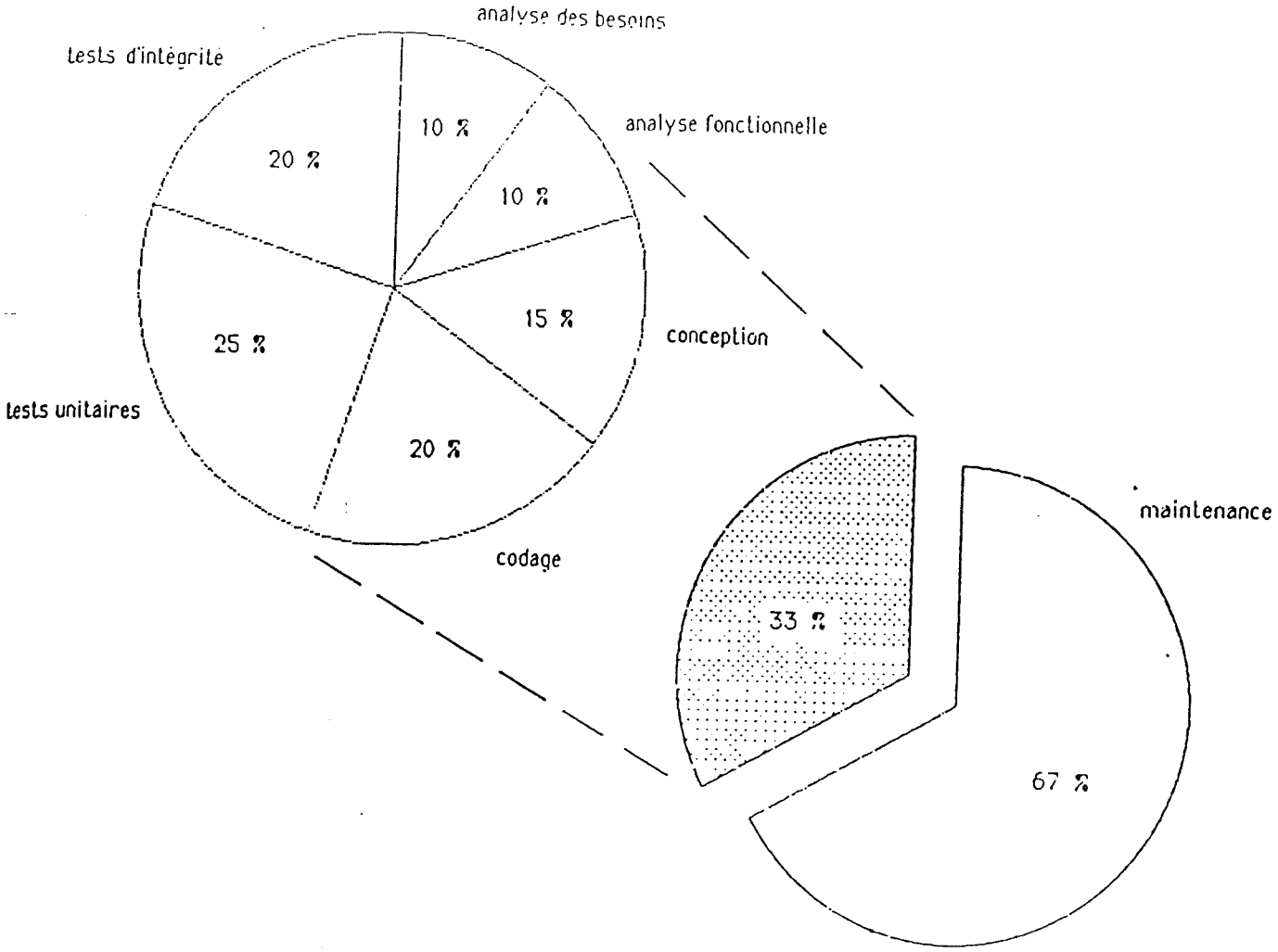


Figure 5 : Répartition du temps alloué aux étapes du cycle de vie d'une application informatique (VAN LAMSWEERDE 88)

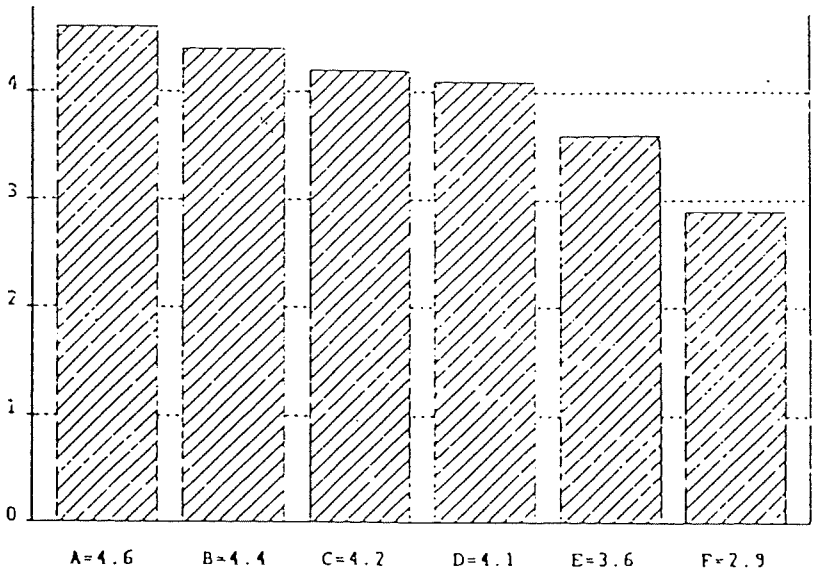


Figure 6 : Facteurs de choix d'un L4G selon une enquête réalisée aux USA à laquelle 80 utilisateurs de ADR ont participés (HENNEAUX 87 p 95)

A. Accroissement de productivité
B. Réduction de la maintenance des applications
C. Capacité à développer rapidement des applications
D. Facilité d'utilisation
E. Réduction du retard informatique
F. Terminer un projet d'envergure dans des délais stricts

5° Intérêts.

+ influence sur la productivité

Un facteur qui influence la productivité du programmeur est la dimension du programme. La productivité sera plus basse avec de grands programmes qu'avec des petits. Dès lors,la méthode de conception doit veiller à créer une architecture modulaire dans laquelle aucun module ne croit d'une maniére excessive. On observe aussi une baisse de productivité avec des programmes extrêmement complexes. La méthode de conception doit également essayer de diminuer la complexité des modules confiés aux programmeurs afin d'accroitre leur productivité. Malheureusement,cette méthode n'est pas souvent appliquée et cela conduit à de mauvaises architectures de sorte que la productivité n'est pas optimale et la maintenance devient très coûteuse. A l'heure actuelle, les 2/3 du coût total en moyenne sont consacrés à des travaux de maintenance (figure 5) [VAN LAMSWEERDE 88].

L'intérêt porté à la diminution de la maintenance et à la productivité est confirmé par une enquête réalisée aux USA à laquelle 80 utilisateurs de ADR (Applied Data Research) ont participé [HENNEAUX 87]. Il était demandé aux personnes interviewées de donner un poids, sur une échelle de 0 à 5 , à un certain nombre de facteurs susceptibles d'avoir joué un rôle dans leur décision d'acquisition d'un L4G en général. Les résultats sont représentés à la figure 6 .

Une des raisons principales d'acquérir un langage de quatrième génération (L4G) est d'augmenter la productivité de la réalisation d'applications. Dans certains cas , l'accroissement de productivité a été faible du fait de techniques de management inappropriées ou du fait de la réaction des utilisateurs de ces L4G.

Le gain par rapport à un langage classique dépend de l'application . S'il s'agit d'une application consistant principalement à générer des rapports , à utiliser des menus ou des écrans d'introduction de données , un L4G sera plus efficace. Par contre , s'il y a beaucoup d'opérations logiques (boucles,structures de CASE,...) avec peu de rapports , d'écrans et d'accès à une base de données , l'intérêt est moindre.

Comme il a été vu au point 4° des Considérations théoriques , l'utilisation d'un L4G peut avoir un impact sur le cycle de vie d'un projet informatique. Dans le cas où on applique le même cycle de développement , de meilleurs résultats peuvent être obtenus (figure 7). C'est surtout aux trois dernières étapes que le gain est net (Programmation, ,Tests,Intégration). Les gains sont cependant meilleurs si un cycle de vie plus approprié au langage est adopté. La figure 8 illustre cette différence.

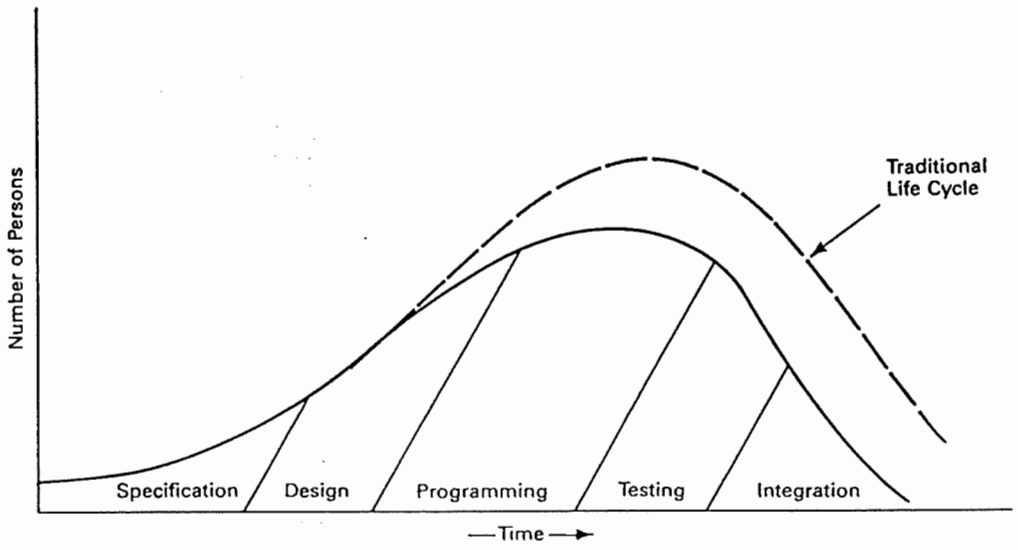


Figure 7 : Effet sur un cycle de vie traditionnel de l'utilisation d'un L4G (MARTIN 85 p 77)

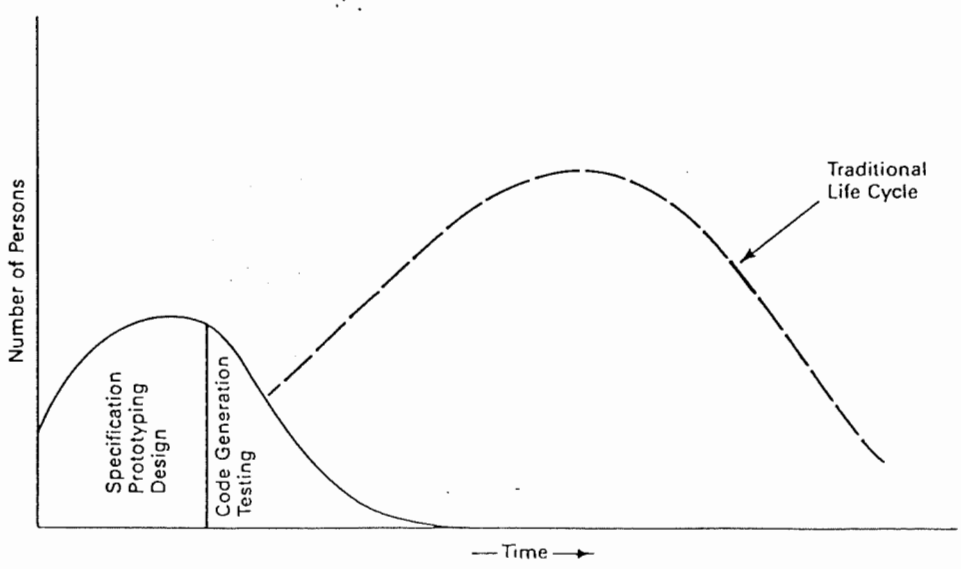


Figure 8 : Effet de l' utilisation d' un L4G sur la productivité par utilisation d'un cycle de vie adapté au L4G (MARTIN 85 p 78)

L'accent est alors mis sur les spécifications et les phases de conception . Dans certains cas, le code est généré à partir de celles-là , leurs élaborations nécessitant donc une très grande précision.

En pratique , si on compare le développement dans un L4G avec le développement en COBOL (ce langage étant le plus utilisé dans la conception de logiciels de gestion) , on observe en moyenne un ratio de temps mis de 1/10 .
La figure 9 montre une comparaison effectuée entre un logiciel développé en COBOL et avec FOCUS [MARTIN 85]. Il est à noter qu'on ne retrouve pas le rapport 1/10 dans le cas présent. Une étude comparative a été réalisée à l'université d'Auckland par le Dr Rudolph. Le résultat de celle-ci est representé à la figure 10 . Les deux langages comparés sont le COBOL (PL/1) et le LINC de Burroughs. Le ratio entre les efforts de développement croit avec la taille de l'application [MARTIN 85].

Non seulement les L4G peuvent augmenter la productivité de manière quantitative avec ou sans modification du cycle de vie d'un projet mais ils peuvent avoir un impact sur la méthodologie de travail. Ainsi , beaucoup de projets de faibles ou de moyennes ampleurs peuvent être confiés à une personne. Et c'est généralement avec de tels projets que les meilleurs résultats sont obtenus.

Cependant, on observe de grandes variations dans la productivité d'une personne. Il peut y avoir une différence de 1 à 10 entre les bons et les mauvais programmeurs . Les projets confiés à une ou à deux personnes étant plus fréquents avec les L4G , d'excellentes performances par les individus sont plus typiques dans ce monde des L4G.

Une variété de facteurs , autre que le langage lui-même , affectent la productivité :

- le temps de réponse au terminal (devant être inférieur à la seconde)
- une machine dédiée au développeur
- éditeurs graphiques
- des outils de modélisation de données (dictionnaires,outils graphiques,...)
- l'absence d'interférence (coups de téléphone , visites de collègues,...)

A coté de cela , il faut également considérer la complexité de l'application à réaliser. Les catégories de L4G se différencient ici. Les L4G procéduraux permettent de

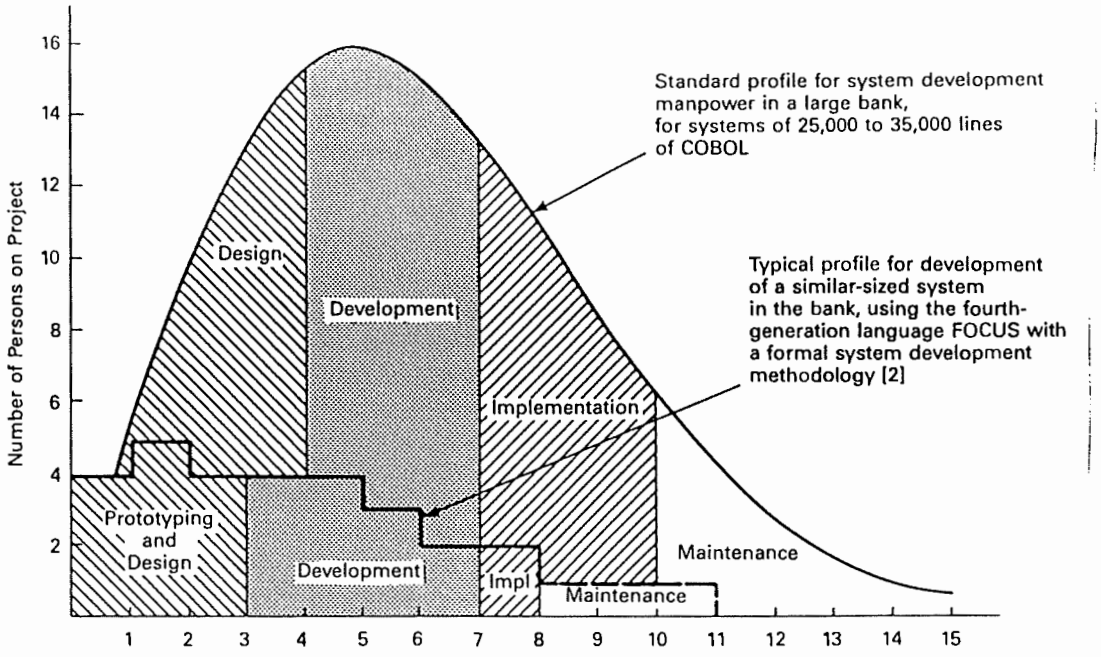


Figure 9 : Comparaison de l'effort de développement de COBOL et de FOCUS dans une méthodologie de développement classique (MARTIN 85 p 80)

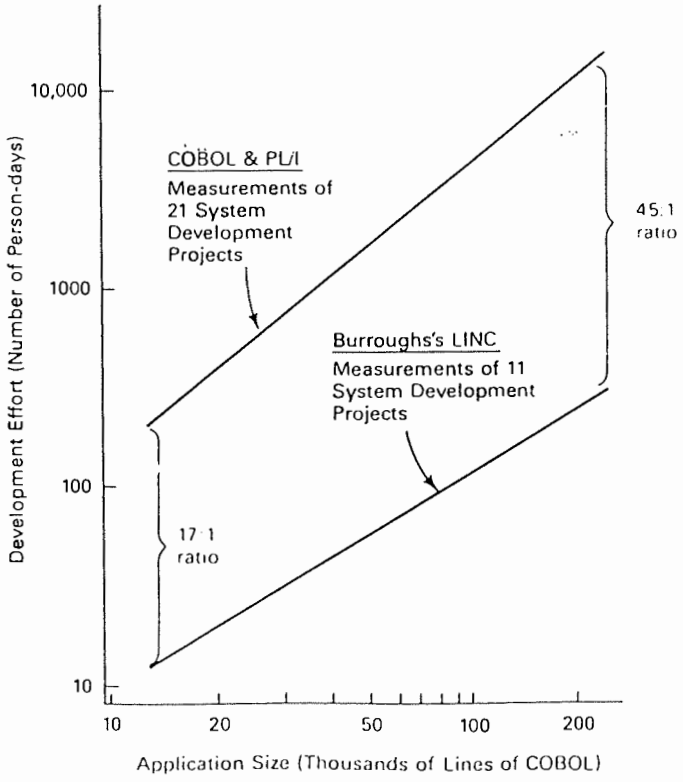


Figure 10: Mesures de productivité de développement faite par l'Université d'Auckland visant à comparer le développement avec le langage COBOL et le LINC de Burroughs (MARTIN 85 p 85)

réduire substantiellement les efforts de développement.

Cependant , les langages non procéduraux offrent une réduction beaucoup plus nette pour des applications peu complexes. Par contre , pour des applications complexes , ces derniers deviennent totalement inefficaces , même par rapport aux langages de troisième génération ! Il convient donc d'utiliser un L4G qui associe de manière appropriée des facilités non procédurales (écrans,accès faciles à une B.D.) et des composantes procédurales. La figure 11 illustre ce qui vient d'être dit [MARTIN 85].

Un certain nombre d'expériences non concluantes ont été rapportées . Quelques raisons à cela peuvent être avancées :

* Plus d'apprentissage est nécessaire : l'organisation n'a pas investi assez d'argent dans l'élaboration d'une équipe avec suffisamment d'habileté et de pratique.

* Certains générateurs d'applications sont limités dans ce qu'ils peuvent générer.

* Un changement insuffisant dans les techniques de management et de contrôle : des contrôles appropriés dans un langage de troisième génération peuvent entrainer la perte de la capacité de développement rapide.

* La non utilisation d'une méthodologie de conception appropriée pour le L4G.

* La non utilisation des particularités interactives de prototypage : il s'agit d'une sous-utilisation des outils offerts.

* L'utilisation d'un L4G qui n'a pas les capacités requises pour l'élaboration de systèmes complexes.

L'augmentation de productivité qui peut apparaître avec ces langages et qui intéresse à juste titre les responsables informatiques peut cacher certains problèmes. Il y a en effet un risque d'incohérence,voir de chaos dans des routines non cataloguées et des données incompatibles. L'importance de celle-ci est accentuée par le fait que les utilisateurs peuvent , dans certains cas , développer leurs propres applications.

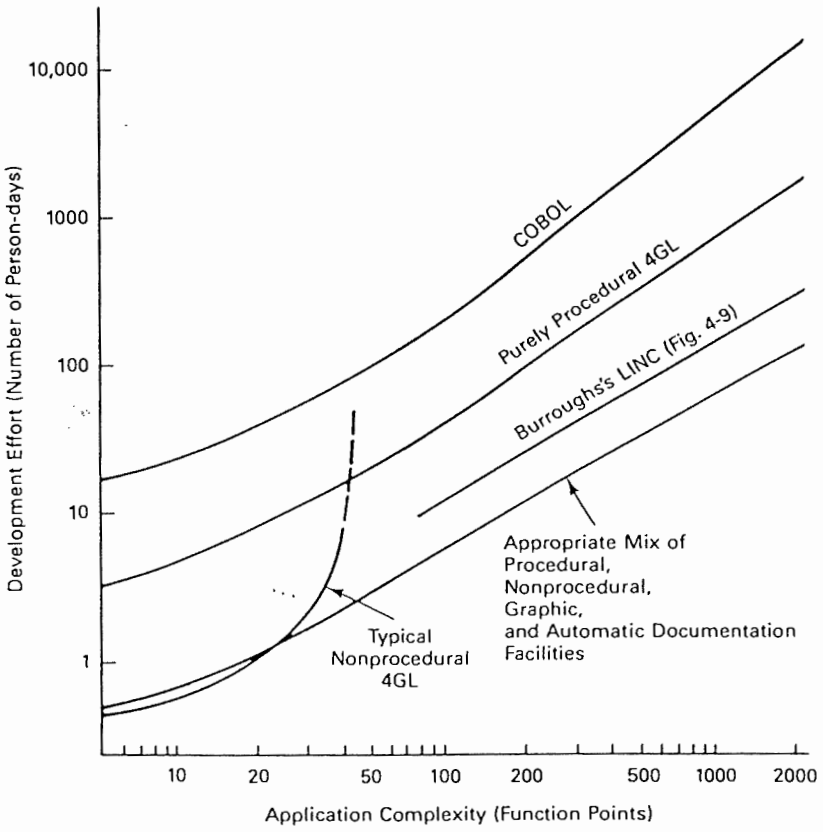


Figure 11: Evolution de l'effort de développement en fonction de la complexité des applications à développer pour le COBOL et pour différents types de langage de quatrième génération (MARTIN 85 p 89)

+ facteurs humains

La prise en compte des facteurs humains a un certain nombre d'effets [MARTIN 85]:

- cela rend les langages faciles à utiliser ce qui encourage une plus grande part des gens à les utiliser. La création d'applications nécessite des analystes plutôt que des programmeurs. Les utilisateurs finaux sont particulièrement intéressés et il est nécessaire que ceux-ci utilisent directement les ordinateurs avec peu de temps et de patience.

- les L4G doivent être "user-friendly" , non menaçants dans leurs messages d'erreur , faciles à apprendre et faciles à se souvenir.

* Structure du dialogue :

Etant donné l'orientation que prennent les L4G vers les utilisateurs finaux, il est nécessaire que celle-ci se reflète dans la structure du dialogue . En effet , les utilisateurs finaux ne travaillent à leurs terminaux qu' occasionnellement. Le taux auquel ils oublient les termes mnémoniques (termes-clés se voulant faciles à retenir !) , les séquences fixées , etc ... est nettement plus élevé que la plupart des programmeurs ne le réalisent , ces derniers travaillant fréquemment à leur terminal.

De plus, il faut permettre à l'utilisateur d'effectuer un retour en arrière (option BACK et/ou ESCAPE) à tout moment.

Les messages d'erreur doivent également être bien élaborés. Plutôt que de donner le type d'erreur sous forme d'un numéro auquel l'utilisateur se réfèrera dans le manuel d'utilisation , il est préférable d'expliquer le plus clairement et de la manière la plus neutre le problème . Par plus clairement signifie sous forme d'une phrase compréhensible et expliquant exactement où se trouve le problème. De la manière la plus neutre veut dire qu'il faut éviter les messages rudes du genre "ILLEGAL FUNCTION" ou "FATAL ERROR" ...

* Problème du "mapping conceptuel" :

Un utilisateur final réfléchit sur des problèmes d'une manière conceptuelle qui est différente de la manière dont l'ordinateur les aborde. D'une manière ou d'une autre , la terminologie doit être traduite d'un monde vers l'autre.

Dans le passé , un analyste faisait cette traduction. Maintenant , dans bien des cas , on aimerait qu'un utilisateur avec peu d'entrainement sur ordinateur fasse celle-ci avec l'aide d'un logiciel. On parle de ceci en terme de problème du "mapping conceptuel".

Le défi des langages de quatrième génération est d'aller aussi loin que possible en permettant à l'utilisateur d'instruire la machine sans pour autant utiliser une syntaxe étrangère et des termes mnémotechniques. Ce défi n'est réalisable que si la plus grande partie du "mapping conceptuel" peut être réalisé par le logiciel.

* Arguments contre l'approche "user friendly" :

Un développeur d'application professionnel peut apprendre les termes et les séquences requises. Ceci constitue un premier argument. Cependant , il est nettement préférable que la personne se concentre sur les problèmes les plus importants , c'est à dire ceux liés à la sémantique plutôt que de dépenser ses efforts sur des problèmes touchant à la syntaxe.

Un autre argument a été avancé. Cette approche est plus exigeante en capacité de cycles machine et en terme d'exigence mémoire. Cet argument était surtout valable avant les progrès réalisés à ce niveau , depuis une quinzaine d'années.

+ prototypage

* intérêt

L'élaboration d'applications complexes nécessite davantage le prototypage que bien des systèmes , car il y a beaucoup plus à apprendre d'une expérience opérationnelle et beaucoup de changements ont grande chance d'être opérés. L'utilisateur final , en effet , sera rarement satisfait de la première version proposée par le programmeur. Dès lors , au moment où l'informaticien fait la première démonstration de son programme , il doit s'attendre à un certain nombre de remarques émanant de l'utilisateur final. Il est rare que ces remarques concernent un fonctionnement intrinsèquement incorrect du programme ; elles se rapportent plus généralement à une mauvaise adéquation du programme aux attentes de l'utilisateur .

Dès lors , il s'agit de disposer d'outils permettant d'ajuster rapidement le programme aux besoins "revus et corrigés " de l'utilisateur final. La plupart des L4G revendiquent cet atout : autoriser une grande souplesse en matière d'adaptation rapide de programmes existants.

Le système obtenu par un cycle de développement classique est en quelque sorte un prototype , bien que n'étant pas considéré comme tel , mais il a les imperfections de celui-ci. Celles-ci sont chères à corriger et restent souvent dans le système conduisant au bout du compte à une maintenance coûteuse.

Jusqu'au début des années 80 , le coût de la programmation d'un prototype revenait à peu près au même coût que le programme final . Les L4G permettent de créer des prototypes à un coup relativement bon marché. Des écrans peuvent être obtenus rapidement , des rapports et le dialogue qui sera proposé peut être perçu à l'avance par l'utilisateur qui pourra y apporter ses modifications et souhaits. Avec un système opérationnel , la perception de l'utilisateur de ce qu'il désire change toujours. De plus , cela permet de s'assurer que l'analyste et l'utilisateur parlent bien de la même chose [MARTIN 85].

* pourquoi convertir un prototype

  - pour une question de performance (sera discuté au point 6°)

  - pour des questions de sécurité , de contrôles d'intégrité , de protection contre les incidents , de fiabilité et pour pouvoir effectuer des audits.

  - pour obtenir la capacité d'avoir des B.D. plus grandes , plus de terminaux , un meilleur réseau ou des volumes de transaction supérieurs.

* types de développement d'application

  Il en existe trois types :

  1 . le cycle de vie classique

  2 . l'utilisation d'un L4G comme outil de prototypage et ensuite le codage dans un langage de troisième génération

  3 . l'utilisation d'un L4G pour développer l'entièreté de l'application

Dans le troisième cas , un utilisateur final et un analyste peuvent travailler de concert pour la création de l'application.

Pour beaucoup d'applications , la troisième approche peut être utilisée mais pas pour toutes. Dans bien des cas , il n'est pas nécessaire d'envisager le problème de l'efficience machine car les programmes ne tournent pas fréquemment . Seulement une faible proportion sont des applications à gros volume. Dans ce cas , une efficience élevée peut être atteinte de trois manières [MARTIN 85]:

1 . Certains L4G sont conçus pour atteindre une efficience machine élevée.

2 . Certaines routines étant "mangeuses de temps" , on peut les isoler et les programmer dans un langage plus efficient.

3 . L'application peut être entièrement reprogrammée.

* Prototypage partiel

Ce type de prototypage peut être plus facile à utiliser et une variété de formes existe (prototypage du dialogue , de l'entrée des données , des rapports , des calculs et des opérations logiques , ...).

* Conséquences du prototypage

a) Quand le prototypage devient très facile , l'organisation responsable du développement ou les utilisateurs terminaux eux-mêmes peuvent créer des systèmes pilotes et les essayer à volonté pour voir si d'autres utilisateurs finaux les apprécient. Dans certaines organisations , un tiers des projets pilotes sont abandonnés.

Par cette technique , les concepteurs de systèmes peuvent découvrir les demandes et raffinements que les utilisateurs utiliseront réellement.

b) Le cycle de développement , comme il a été vu au point 4° , se trouve modifié. Un cycle différent peut être celui de la figure 3.

c) Il nécessite l'utilisation d'un langage de prototypage. Généralement , celui-ci est réalisé avec le même langage utilisé pour le développement final. Malgré les différences existant entre un langage de prototypage et un langage de développement (ce dernier nécessitant de bonnes performances machine , un système de gestion de base de données traditionnel,...) , il est intéressant d'utiliser le même langage pour trois raisons :

Prototypage	
Avantages	Inconvénients
l'utilisateur n'est pas surpris	cache des données à l'utilisateur
prise en compte des exigences de l'utilisateur	utilisateur moins raisonnable
visualiser l'état d'avancement du travail	limite le niveau de réflexion de l'utilisateur
résultat plus proche des exigences de l'utilisateur	

Figure 13: Principaux avantages et inconvénients du prototypage (HENNEAUX 87 p 115)

1 . cela permet d'économiser plus de temps de développement

2 . cela autorise une continuité entre le prototypage et le système final

3 . cela rend possible la modification du système final de la même manière que le prototype

Malgré ses nombreux avantages , cette technique rencontre encore beaucoup de résistance de la part des managers qui hésitent à changer leurs méthodes et par manque de connaissance des outils disponibles.

Les principaux avantages et inconvénients du prototypage sont résumés à la figure 13.

+ langages de spécification

Les spécifications telles qu'elles sont fréquement réalisées posent des problèmes :

- elles sont souvent longues et ennuyeuses ce qui a comme conséquence que les managers ne les lisent pas en entièreté.

- elles manquent de précision

- elles sont souvent ambiguës , contradictoires et incomplètes

- elles sont souvent mal interprétées des deux côtés (utilisateurs et analystes)

- elles comportent du bruit qui n'apporte rien

Dans les premiers jours de l'histoire informatique , les langages étaient pensés seulement dans le but de la programmation. Ils étaient par conséquent fort proches dans leur syntaxe des instructions d'une machine. L'apparition des langages comme le COBOL , le FORTRAN , l'ADA , etc... a humanisé quelque peu la programmation en introduisant des mots d'anglais mais ça n'a en rien changé l'optique qui était d'écrire des programmes ayant le même "mode de raisonnement" que la machine.

L'écriture des spécifications nécessitent un tout autre langage. D'abord réalisées dans un langage humain qui posait les problèmes signalés ci-dessus , les concepteurs se sont tournés vers des langages de spécification plus

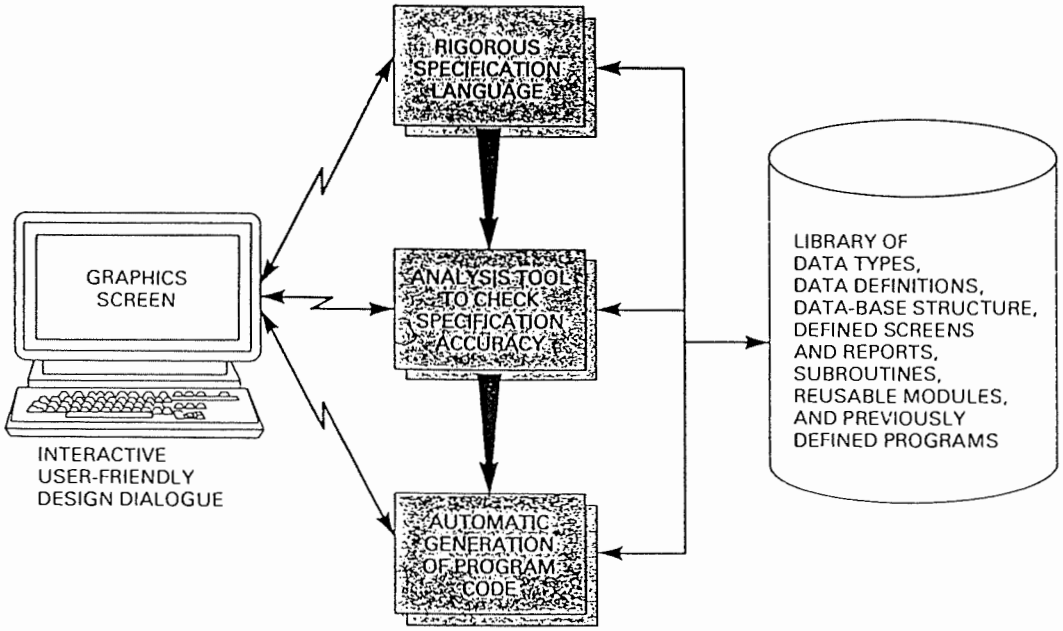


Figure 14: Utilisation d'un langage de spécification pour la génération automatique de code (MARTIN 85 p 323)

rigoureux. Cependant , il existait toujours une dichotomie entre ces deux types de langage.

Actuellement , la conception de langages de spécification qui soient capables de générer du code constitue un pôle de la recherche. Cette capacité nécessite une grande rigueur dans les spécifications , bien plus grande qu'elle n'est souvent présente dans la réalité. Des vérifications sont par conséquent effectuées sur ces spécifications avant la génération du code. La figure 14 illustre le principe.

Contrairement à la voie classique utilisée , un outil de cette sorte permet d'intégrer toutes les étapes de réalisation d'un projet. Les spécifications peuvent être écrites et vérifiées rapidement , ensuite le code généré. On évite donc les erreurs intervenant entre les spécifications d'un cycle classique et le codage qui fait apparaître d'autres erreurs. Vu la cohérence du système évoqué ci-dessus , les erreurs de codage sont évitées puisque celui-ci est issu de manière formelle et automatisée.

Pour atteindre ce but , il est nécessaire d'être rigoureux. Cette rigueur implique l'utilisation de règles mathématiques. Celles-ci devront être complétement cachées car beaucoup de managers et d'analystes sont terrifiés par les mathématiques. Un défi majeur se présente donc à l'industrie informatique qui pour atteindre cet objectif doit tenir compte des trois aspects fondementaux :

1. la rigueur des spécifications
2. le caractère "user friendly"
3. l'étendue du langage à différents types d'application

\+ adaptation

S'orienter vers un L4G dépend d'un certain nombre de paramètres :

- certains L4G sont adaptés pour les utilisateurs finaux , d'autres pour des professionnels du développement
- si l'environnement est changeant , c'est à dire que le système risque de devoir être modifié (même au niveau des spécifications) , il est préférable de s'orienter vers un L4G
- beaucoup de L4G sont destinés seulement pour des applications commerciales. Pour le calcul scientifique , les langages de troisième génération sont mieux adaptés

- un grand volume de transactions est un argument défavorable au point de vue performance (voir point 6° qui suit)
- le degré de complexité de l'application
- l'importance de la B.D.
- le type d'ordinateur utilisé (PC,mini,mainframe ou une combinaison de ceux-ci)
- l'accés via un réseau
- l'utilisation ou non du prototypage : dans l'affirmative, un L4G est parfaitement adapté

L'environnement dans lequel on se trouve a donc une importance fondementale dans l'adaptation ou non d'un L4G.

6° Performances.

Par rapport aux langages classiques , les L4G consomment plus de ressources. Il serait intéressant d'effectuer une comparaison chiffrée sur l'impact au niveau des ressources machine entre les langages de troisième et quatrième génération . Les unités de mesure indispensables dans cette comparaison sont les suivants :

- le temps CPU
- la mémoire centrale utilisée
- le temps d'utilisation des canaux pour les E/S
- le temps d'attente
- le temps total de présence en machine

Cette liste est loin d'être exhaustive.

Un langage puissant pour l'utilisateur (par ex : QUERY_BY_EXAMPLE) permet aux utilisateurs de se lancer dans des opérations consommatrices de temps machine. En effet , l'utilisateur peut réaliser une opération de liaison relationnelle portant par exemple sur deux tables de 10000 enregistrements . Dès lors , si plusieurs utilisateurs effectuent des opérations similaires , le temps de réponse des autres sera plus long - ce qui constitue un problème [HENNEAUX 87].

Le concept de performance est très lié à la notion des ressources évoquée ci-dessus. En effet , une dégradation des performances entraine souvent la nécessité d'augmenter les ressources.

Certains programmes écrits dans un langage de quatrième génération offrent des performances moins "bonnes" que les programmes écrits dans un langage classique. Certains L4G n'ont pas été conçus dans le cadre d'une optimisation globale du code produit. Dès lors , une application développée dans un L4G peut avoir un temps d'éxécution non acceptable. Pour résoudre ce problème , un spécialiste de ce L4G devra optimiser toute l'application , si c'est possible [MARTIN 85].

Il faut signaler qu'il existe des L4G qui sont conçus pour donner de bonnes performances machine , parfois meilleures que les applications COBOL.Ceci est rendu possible par le codage de certains blocs en langage machine.

Les problèmes de performance peuvent aussi survenir lorsque le L4G est interprété plutôt que compilé.

3.2 LE LOGICIEL POWERHOUSE.

1. INTRODUCTION.

Le logiciel Powerhouse, développé par la firme COGNOS il y a quelques années, est un logiciel de programmation dit de "quatrième génération".

Il a ainsi bon nombre des caractéristiques énumérées au chapitre précédent et propres à ce type de langage.

On peut le considérer comme un système intégré de développement d'application.

Créé pour tourner uniquement sur mini-ordinateur ou mainframe, il est de plus étroitement lié au matériel VAX de Digital (récemment, une version simplifiée tournant sur micro-ordinateur a été commercialisée).

Le système Powerhouse permet de développer entièrement une application sous forme de prototype très rapidement construit et de modifier celui-ci beaucoup plus facilement qu'avec un langage de troisième génération, en fonction des souhaits des utilisateurs.

Il suggère ainsi une démarche interactive entre l'analyste (devenu aussi programmeur de l'application) et l'utilisateur.

Dans ce chapitre, nous aborderons les composants et l'organisation du logiciel Powerhouse. Ses traits caractéristiques seront dès lors mis en évidence.

Nous décrirons ensuite le domaine d'application vers lequel il s'oriente tout particulièrement.

2. DANS QUELLE CATEGORIE DE L4G POWERHOUSE SE CLASSE-T-IL ?

D'après les grandes catégories de langage de quatrième génération dégagées au chapitre précédent, nous pouvons dire que Powerhouse est un langage de programmation de haut niveau non procédural avec des possibilités procédurales pour certains aspects.

Il est en grande partie non procédural en ce sens qu'il cache le "comment" au profit du "ce qu'il faut faire".

Il est, par exemple, inutile de gérer normalement l'ouverture et la fermeture des fichiers ou bien la lecture et l'écriture d'enregistrements (des options pour le faire sont cependant disponibles).

Des instructions uniques du type

"ACCESS file" (pour l'utilisation d'un fichier)
ou
"OUTPUT file ADD" (pour l'ajout de records dans un fichier)

s'occupent de toute une série d'opérations purement physiques liées à ce genre d'opération.

L'aspect procédural complémentaire offre une possibilité de traitement plus complexe et étend la gamme d'applications réalisables.

Il consiste en des expressions habituelles aux langages procéduraux, du type

"IF condition THEN action ELSE action"

ou d'expressions plus ou moins optimisées du type

"WHILE RETRIEVING file" (permet de parcourir tous les enregistrements d'un fichier sous une certaine valeur de clé)

Enfin, il faut noter que Powerhouse se classe également parmi les langages interactifs. Interaction "on line" disponible pour la programmation, mais pas obligatoire (un éditeur existe). Interaction plein écran pour la saisie des données du dictionnaire (v.point suivant).

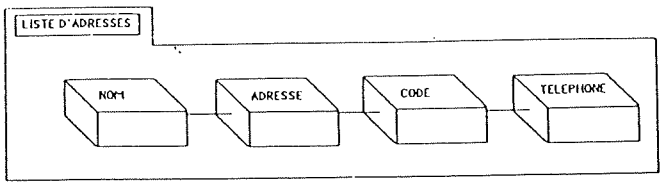
Dans toute application Powerhouse, on distingue comme type de donnée :

* LES ELEMENTS :

Ce sont les différents éléments pouvant intervenir dans la description d'un fichier.
Pour chacun d'entre eux, il faut préciser leur nom, leur type (caractère,entier,...), leur format, leur taille,...etc.
Chaque élément ainsi décrit, une fois pour toute, peut alors servir pour la description d'autant de fichiers que l'on désire.

* LES RECORDS :

A partir des éléments décrits ci-dessus, on spécifie la structure d'un record d'un fichier.
On peut faire de certains éléments des clés (uniques ou répétées) de fichiers.
A ce stade, on parle de "structure de record".



Un ensemble de records ayant cette structure est contenu dans un fichier.

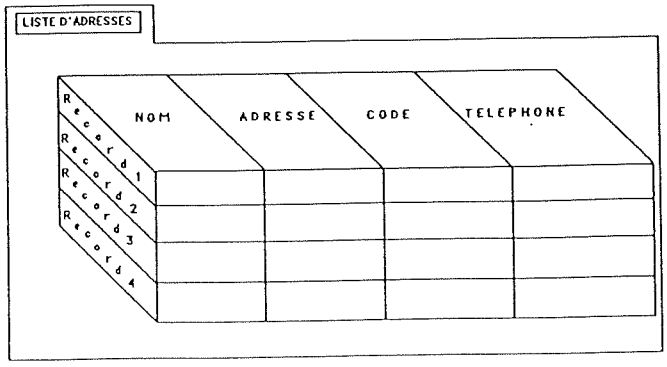
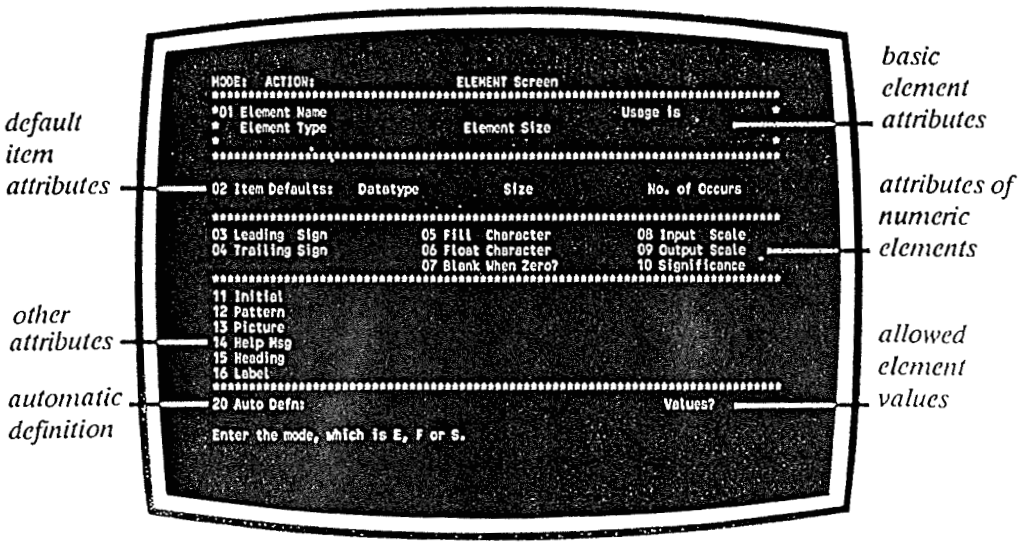
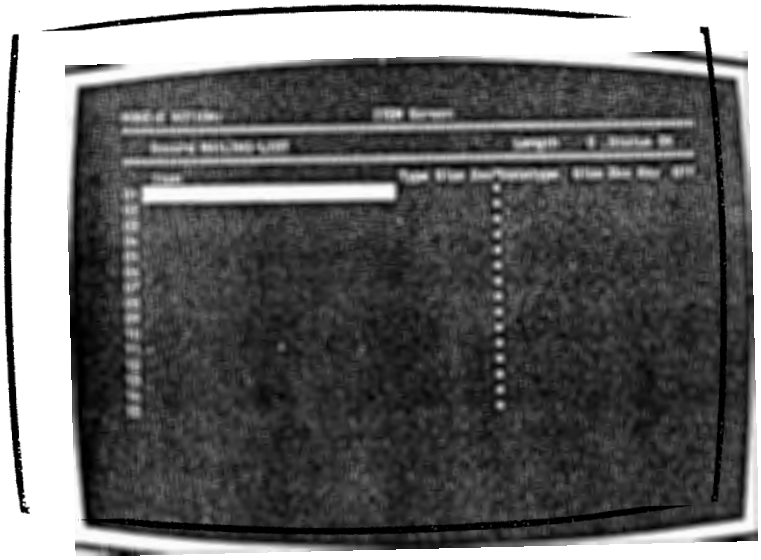
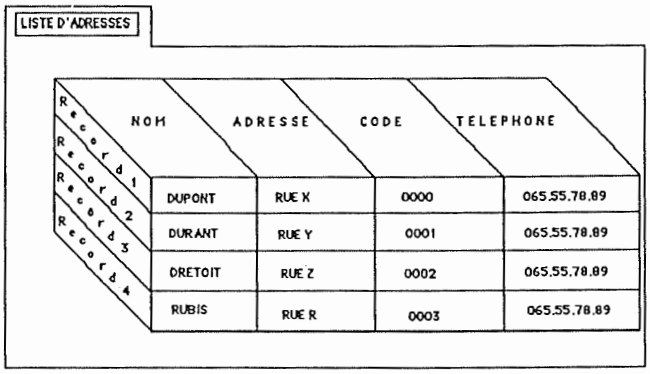


Figure 15 : Ecran proposé pour l'entrée de nouveaux ELEMENTS et en-dessous, pour l'entrée d'ELEMENTS (items) de records de fichiers .
(COGNOS 89 PhD chap.1)





Enfin, les valeurs sont ajoutées via les composantes QUICK et QTP de Powerhouse (v.point suivant).



La saisie de toutes ces informations (exeptées les valeurs proprement dites) se fait interactivement via des écrans proposés au développeur.

Le type d'écran proposé pour l'entrée de nouveaux "éléments" est représenté à la figure 15 .

L'utilisateur/développeur est invité à compléter successivement les zones prédéfinies sur l'écran.

Il est cependant possible de se positionner sur une zone bien précise, si on le souhaite, en spécifiant son numéro.

N.B. :

Il faut noter que tout écran Powerhouse (c'est-à-dire préconçu ou créé par le développeur) est basé sur le principe suivant.

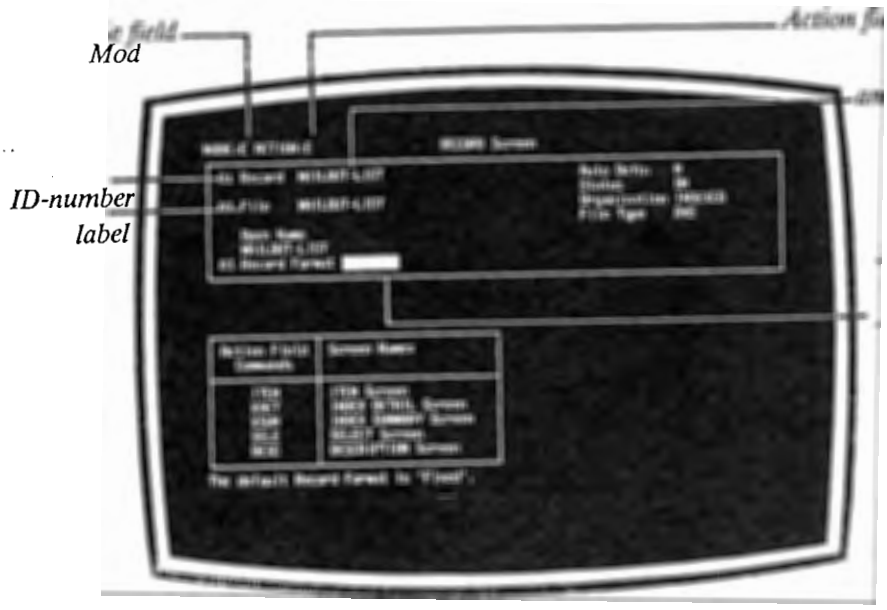
Dans le coin supérieur gauche d'un écran se trouve une "zone action" par laquelle il faut obligatoirement passer préalablement à toute entrée, recherche ou modification.

Il suffit d'y spécifier le type d'action que l'on désire effectuer sur l'écran pour, ensuite, en spécifier le contenu (MODE powerhouse).

Ainsi, par exemple, en tapant la lettre E dans la zone action, on spécifie le mode ENTRY c'est-à-dire l'entrée de nouvelles données ou descriptions.
La lettre F signifiant quant à elle le mode FIND, c'est-à-dire le désir d'une recherche d'une description ou d'une information particulière (que l'on précisera dans l'écran lui-même).

Une fois la zone action remplie, l'utilisateur peut visiter les zones apparaissant sur l'écran en fonction de ce qu'il veut faire (v.ex.ci-dessus).

En ce qui concerne les records de fichiers et les fichiers eux-mêmes, Powerhouse propose, pour leur description, ce type d'écran



Pour les fichiers, plusieurs types d'organisation sont possibles. L'organisation à accès direct, séquentielle ou séquentielle indexée.

Avec cette dernière, il est possible d'utiliser une ou plusieurs clés (primaire ou secondaires), uniques ou répétées, à indexation ascendante ou descendante.

De plus, la fusion de plusieurs champs pour constituer une seule clé est réalisable et permet, si on l'a spécifié, d'utiliser chacun de ces champs eux-mêmes comme des clés. Il existe un système de cache pour les parties de la clé qu'on voudrait momentanément ne pas utiliser.

Il est cependant évident que l'utilisation de clés coûte cher en place mémoire pour le stockage des tables d'index et cela d'autant plus que la clé est longue ou qu'il y en a beaucoup !

Une fois la description des fichiers effectuée, Powerhouse maintient cette déclaration dans des fichiers (extension .FDL).

Il suffit alors de créer physiquement le fichier pour en disposer (extension .DAT).

Ceci a l'avantage que, lors de l'effacement du fichier .DAT, sa description figure toujours dans le dictionnaire et peut donc être réutilisée pour créer un nouveau fichier, modifié ou non, en fonction de la description qu'on en a faite (description qu'on a d'ailleurs pu changer au préalable).

Cette facilité de modifications des descriptions de fichiers et leur centralisation font du dictionnaire Powerhouse un outil puissant pour la gestion des données.

En conclusion, on peut souligner le gain en effort de codage qu'apporte le dictionnaire.

En effet, dans un programme écrit avec un langage traditionnel comme COBOL, 40 à 60 % des lignes de code décrivent les données.

Le programme doit indiquer comment les données sont stockées, comment on les accède et comment il faut les présenter. Chacun de ces attributs concernant les données sont à déclarer dans chaque programme, individuellement !

Avec un dictionnaire, on décrit les données une seule fois.

Les attributs sont spécifiés ou modifiés à un seul endroit plutôt que dans les différents programmes. Ceci permet une standardisation à travers toute une application et réduit les coûts d'implémentation et de maintenance.

B. Les exécutables Powerhouse.

Une fois les déclarations effectuées dans le dictionnaire, on dispose de trois exécutables qui permettront de développer l'application.

L'exécutable "QUICK" est un processeur de transactions interactives "on line". Le "QTP" est un processeur de transaction "de masse", tandis que le troisième exécutable, le "QUIZ", est un générateur d'états imprimés.

Du fait de l'aspect système intégré de Powerhouse, ces composantes traivaillent interactivement. On peut, par exemple, créer des applications basées sur les écrans QUICK qui interagissent avec le QUIZ et le QTP.

De plus, ils possèdent certaines structures intermédiaires utilisables en commun (sous-fichier) et une syntaxe souvent similaire.

Enfin, il faut garder à l'esprit que ces trois exécutables se servent directement et constamment du dictionnaire PhD. Celui-ci tient donc une place tout à fait essentielle dans le système.

1° L'exécutable "QUICK".

Ce package est en fait constitué de deux programmes. Le premier, QDESIGN, est utilisé par le concepteur pour créer des écrans, pour modifier des écrans existants ou bien pour développer la totalité de la partie entrée des données d'une application.

Le deuxième est utilisé pour faire exécuter ces écrans à la fois par les opérateurs et par les autres utilisateurs.

La relative simplicité du QUICK doit être attribuée à l'existence du dictionnaire de données (cfr.ci-dessus) qui contient une liste des noms et les caractéristiques de base de chaque fichier et item utilisés dans l'application.

C'est en se référant au dictionnaire que QDESIGN et QUICK peuvent déterminer, par exemple, le type et

la taille maximum autorisée pour une valeur saisie lors de toute entrée. C'est aussi grâce à lui qu'ils peuvent vérifier des caractéristiques facultatives telles que le type d'affichage pour les données, des limites de valeurs acceptables pour une entrée,...etc.

L'élaboration d'un écran qui constitue ce que l'utilisateur voit effectivement de l'application, se fait via l'utilisation d'une syntaxe propre à QDESIGN.

Pour tout écran, il faut préciser :

- son nom : il sert à identifier un écran et permet à d'autres écrans d'appeler celui-ci , qui est alors considéré comme un sous-écran (SUBSCREEN).

- le(s) fichier(s) utilisé(s) : ils doivent avoir été décrits dans le dictionnaire.

- les champs (FIELD) ou zones qui apparaissent sur cet écran : ceux-ci peuvent être des champs de fichier à compléter et déclarés auparavant, ou des champs temporaires permettant la saisie d'information plus générale.

Habituellement, s'ajoute :

- les contrôles tant syntaxiques que logiques devant être effectués sur les données saisies.
  Ce sont des contrôles autres que ceux qui ont été spécifiés dans le dictionnaire et qui sont réalisés uniquement par celui-ci.

- les traitements à effectuer sur les zones de l'écran. Ceux-ci peuvent être spécifiés sous forme d'expression assez brèves (forme non procédurale) ou bien avec des procédures jointes en fin de programme pour l'écran considéré (forme procédurale).

Il existe aussi des procédures prédéfinies par QDESIGN et qui gouvernent les activités de toute entrée, mise-à-jour et effacement de données. Elles peuvent être modifiées ou complétées par le programmeur.

Lors du choix d'une action sur un écran, QUICK passe le contrôle de l'écran à la procédure appropriée jusqu'à la fin de celle-ci.

Par exemple, la procédure prédéfinie ENTRY permet d'effectuer un traitement particulier (qu'on a joint au corps de la procédure) à chaque entrée à l'écran,

la procédure EXIT permet de réaliser une ou plusieurs opérations chaque fois qu'on quitte un écran,...etc.

Un exemple simplifié de conception d'un écran est illustré ci-dessous. Pour plus de détails, on peut se référer au code des écrans (fichiers à extensions .QKS) réalisés pour notre logiciel (cfr. annexe n° 6).

```
SCREEN  Personnel
FILE EMPLOYE
TITLE "Fichier du personnel employe"
FIELD NUMERO-EMPLOYE OF EMPLOYE
FIELD NOM OF EMPLOYE
FIELD PRENOM OF EMPLOYE
FIELD ADRESSE OF EMPLOYE
BUILD
```

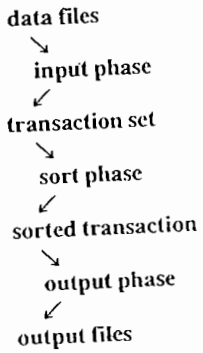
Le QDESIGN a consulté le dictionnaire de données, a déterminé le nom, le type et la taille de chaque champ (item) du fichier Employe et créé un écran permettant la saisie des valeurs.

Toutes les interfaces entre l'écran et les fichiers de données aussi bien que les fonctions de gestion d'écran sont assurées automatiquement.

Une fois l'élaboration de l'écran terminée, il suffit de le compiler (BUILD), par utilisation du QUICK, pour disposer d'un écran sous forme compilée, c'est-à-dire autonome et pouvant être lancé à partir de n'importe quel autre écran (hiérarchie d'écrans).

Enfin, on peut souligner que l'exécutable QUICK interagit avec les deux autres composants (QTP et QUIZ) en ce sens qu'il est possible de lancer des réalisations effectuées à partir de ces derniers au départ d'un écran réalisé par le QUICK.

Figure 16 : Les différentes phases dans une "REQUEST" de QTP.
(COGNOS 89 QTP fig 1.3)



2° L'exécutable "QTP".

Le processeur de transaction "QTP" est destiné au traitement rapide de données de masse ou à la réalisation de travaux plus complexes que ceux effectués par le QUICK.

Il peut ainsi servir aussi bien à la mise à jour journalière, aux backups mensuels qu'au reformattage de fichier ou à la copie sélective de fichier.

Comme le QUICK, il se base sur le dictionnaire (PhD) pour vérifier que les éléments et les records de fichier correspondent bien aux déclarations qui ont été faites.

Ici aussi, la mise sur pied d'un programme QTP nécessite l'utilisation d'une syntaxe propre au QTP.
Les déclarations ou expressions sont non procédurales et se rapprochent de celle du QUICK et du troisième exécutable (QUIZ), ce qui en facilite l'utilisation.

Tout programme QTP comprend, au point de vue code, les structures syntaxiques suivantes:

- le titre du module réalisé

- une ou plusieurs requêtes (REQUEST) qui réalisent un ensemble de traitements liés. Elles sont nettement séparées les unes des autres (on pourrait parler de sous-modules) et sont indépendantes.
  C'est un ensemble de requêtes qui constitue le programme QTP.

Une requête consiste en fait en une lecture de un ou plusieurs fichiers en entrée, habituellement pour mettre à jour un ou plusieurs fichiers en sortie.

On distingue trois phases dans une telle requête : la phase d'entrée, la phase de tri et la phase de sortie (v.fig.16).

Dans la phase d'entrée, QTP effectue une série de transactions sur les fichiers en entrée. Les transactions sont alors triées. Finalement, QTP utilise celles-ci pour mettre à jour les fichiers en sortie.

Les déclarations QTP de tout programme ont donc la structure suivante :

```
RUN Nom-du-programme
REQUEST nom-de-la-requete
     Traitements en entrée
     Traitements de tri
     Traitements de sortie
REQUEST nom-de-la-requete
     Traitements en entrée
     Traitements de tri
     Traitements de sortie
BUILD
```

Cette structure d'un QTP encourage donc le concepteur à diviser des traitements importants en plusieurs petits modules reliés et plus simples.

A titre d'exemple, voici le code d'une "request" élémentaire :

```
RUN programme
REQUEST extraction
ACCESS Clients                  Phase Input
SELECT IF region = 14
SORT ON nom-de-client           Phase Sort
OUTPUT Clients-region ADD       Phase Output
```

Explications :

Dans la phase Input de la requête, QTP lit tous les enregistrements du fichier Clients.

On peut remarquer à ce niveau qu'un accès (ACCESS) peut spécifier plusieurs fichiers à la fois. Une instruction du type "ACCESS fichier1 LINK TO fichier2" signifie la lecture de tous les enregistrements du fichier1 ainsi que tous ceux du fichier2 qui leur sont reliés (présence de champs identiques dans les deux fichiers).

C'est-à-dire les enregistrements vérifiant une même valeur de la clé primaire (par défaut) pour les deux fichiers.

Cette étape préalable d'accès à un certain nombre d'enregistrements de fichier est nécessaire pour une sélection ultérieure (facultative).

Celle-ci se fait par utilisation d'une fonction SELECT (sélection séquentielle) pour vérifier une condition particulière de sélection à travers l'ensemble des enregistrements (ici IF region = 14), ou bien en utilisant une fonction CHOOSE pour le même type d'opération mais effectuée cette fois par l'intermédiaire d'une clé (sélection indexée).

La phase suivante est facultative. Elle permet un tri d'enregistrements selon la valeur d'un ou plusieurs

champs des fichiers impliqués dans la phase Input.
Dans l'exemple, le tri est réalisé selon le nom du client.

Enfin, la phase Output termine la "Request". Elle permet la mise-à-jour des enregistrements des fichiers utilisés (réécriture, ajouts de nouveaux records,...etc.) ou d'autres fichiers.

Il peut exister plusieurs instructions OUTPUT, celles-ci devant préciser le fichier de sortie ainsi que les différents champs à garnir.

La phase Output de l'exemple a ainsi pour fonction d'ajouter (ADD) dans le fichier Clients-region les enregistrements sélectionnés dans les phases précédentes.

Une fois le texte du module terminé, il suffit de le compiler par l'exécutable QTP pour disposer d'une version compilée, autonome. Ceci permettra de pouvoir utiliser ce module de traitement dans des écrans développés avec le QUICK.

En guise de conclusion sur QTP, il faut savoir que toute méthode de modification de données sur fichier doit être sévèrement contrôlée.

Il est conseillé de tester les QTP réalisés sur des échantillons de données afin de voir s'ils opèrent bien selon leurs spécifications.

De plus, il existe des moyens (LOCKING,..etc.) d'empêcher des mises à jour simultanées qui provoquent les plus importantes pertes d'intégrité des données ! Ils sont gérés automatiquement par QTP.

3° L'exécutable "QUIZ".

Ce troisième exécutable se distingue nettement des deux autres par le fait qu'il est uniquement destiné à l'obtention de rapports sous forme texte, sur le contenu des fichiers.

Du point de vue syntaxique, il se rapproche de la syntaxe nécessitée sous QTP. C'est ainsi que l'on trouve des instructions du type "ACCESS file","SORT ON","SELECT IF".

Pour l'élaboration de tout rapport, il faut au minimum spécifier :

- les fichiers à consulter
- les champs de fichiers (item) qu'il faut mettre dans le rapport
- l'initialisation ou le lancement de la production de rapport.

Par exemple, pour établir une liste du contenu de cinq champs du fichier "Employe", il faut entrer :

```
>ACCESS EMPLOYE
>REPORT EMPLOYE PRENOM NOM VILLE DATE
>GO
```

La forme du rapport obtenu sera la suivante :

```
89/08/01          EMPLOYES                    PAGE 1

      NUMERO     PRENOM     NOM          VILLE        DATE
      EMPLOYE

      000001     OLIVIER    DUPUIS       NAMUR        87/01/01
      000002     PHILIPPE   LAURENT      LIEGE        87/04/08
      000003     MARTINE    BODDEN       SPA          86/09/11
      000004     MICHEL     LAMBOT       BRUXELLES    85/11/06
      000005     OLIVIER    LALOUX       BINCHE       89/05/30
```

On constate que le rapport contient l'information formattée automatiquement de la meilleure façon possible avec la date du jour, le titre, le numéro de page et les entêtes de colonne.

La plus grande part de la simplicité du QUIZ est due en réalité à l'interfaçage automatique avec le dictionnaire de données. C'est celui-ci qui fournit l'information sur les fichiers et leurs champs, utilisée dans le rapport. C'est en se référant au dictionnaire que le QUIZ peut déterminer le type, la taille et les spécifications du format de chaque champ.

A côté de ces instructions de base, le QUIZ permet d'accéder à plusieurs fichiers à la fois pour générer un rapport.

Le premier fichier accédé est le fichier primaire.Les fichiers qui lui sont reliés, sont spécifiés par l'option LINK TO. Ces liens sont construits automatiquement par le QUIZ.

Par exemple, le texte suivant montre l'utilisation de plusieurs liens en cascade :

```
ACCESS EMPLOYE
    LINK TO DEPARTEMENT
    LINK TO DIVISION
    LINK TO POSITION
```

Le système essaie d'établir une relation entre tous les fichiers dénommés dans la liste des accès de la façon suivante :

1.- pour le fichier DEPARTEMENT, le QUIZ recherche le premier fichier accédé (EMPLOYE) et essaie d'y trouver un nom de champ qui correspond à un nom d'une clé du fichier DEPARTEMENT.

2.- Ensuite, pour le fichier DIVISION, il recherche d'abord, parmi les employés, un champs qui correspond à une clé du fichier DIVISION et puis, si nécessaire, du fichier DEPARTEMENT.

3.- enfin, pour le fichier POSITION, il recherche dans les fichiers nommés dans la liste ACCESS à partir du premier fichier (EMPLOYE), jusqu'à l'avant dernier fichier (DIVISION) afin de trouver un champ qui correspond à une clé du fichier POSITION.

rem : les champs qui correspondent ne doivent pas nécessairement être eux-mêmes des clés, mais leur nom doit être identique.

En conclusion, l'exécutable est assez souple à manipuler. Il offre une gamme variée de types de rapport, allant du rapport de base à des rapports complexes faisant intervenir plusieurs fichiers.

Différentes possibilités de présentation sont offertes au programmeur : rapport avec entête, notes de fin de page, formattage des données en colonne, avec espacements, ...etc.

4. TYPE DE DOMAINE D'APPLICATION DE POWERHOUSE.

En tant que langage de quatrième génération, Powerhouse permet d'augmenter la productivité et la créativité des informaticiens mais aussi d'utilisateurs néophites.

L'accès au système informatique par les utilisateurs de tous niveaux se voit facilité.
Que cela soit au niveau de l'entrée des données, des requêtes "on line" ou pour la génération de rapports.

Ce système intégré de développement d'application peut être considéré comme l'outil idéal pour la réalisation d'un certain type de projet informatique.

Il s'oriente vers des applications nécessitant un développement rapide et aisé. Ainsi, il convient particulièrement bien pour le traitement de données commerciales et leur gestion, pour des applications orientées base de données, pour la gestion de rapports, la comptabilité ou les interfaçages avec des appareils de mesure pour la saisie et le stockage des données, ou encore pour des traitements en batch.

Il n'est pas du tout adapté à des applications scientifiques surtout du fait de ses moins bonnes performances dans le domaine calcul ou opérations complexes.

A propos de l'utilisation de Powerhouse, il faut signaler qu'il n'est pas toujours facile de maîtriser les finesses de ce type de langage, ne fût-ce que pour la compréhension de certaines fonctions offertes pour lesquelles il n'y a pas d'explication détaillée.

Ainsi, certains aspects de fonctionnement (aspects plus physiques) sont cachés au développeur.
Il est alors difficile de comprendre la démarche suivie et les fonctionnalités exactes, surtout quand la documentation est insuffisante.

Cela se remarque lors de la conduite des tests car il est parfois difficile de situer une erreur ou bien d'identifier la structure qui la provoque; une structure complexe cache toute une série d'opérations distinctes dont il faudrait connaître les résultats.

De plus, il semble préférable et plus facile d'apprendre ce type de langage de quatrième génération sans se baser sur des principes ou des concepts propres aux programmeurs ayant utilisé un langage de troisième génération.

Il semble plus facile pour un débutant de se familiariser avec ce langage que pour un ancien programmeur qui peut difficilement se défaire de ses vieilles habitudes de programmation.

Chaque langage de quatrième génération a ses propres constructions et sa propre "philosophie" qu'il est bon d'assimiler indépendamment de tout contexte de langage de troisième génération. Il faut pouvoir utiliser complètement les facilités du langage et en avoir une approche appropriée.

Enfin, du point de vue performance, il faut être conscient du type de travail que l'on peut demander ou non à tel type de L4G.

Comme énoncé précédemment (v.chapitre 3 point 6°), les performances sont fortement liées au type d'application à réaliser et à la disponibilité d'utilitaires plus ou moins optimisés.

Par ailleurs, bien que n'entrant pas dans le cadre du mémoire, il serait intéressant d'effectuer des tests de performance pour voir dans quels secteurs Powerhouse est plus ou moins performant par rapport à un langage comme le COBOL.

3.3 STATISTIQUES.

1. INTERETS DES STATISTIQUES EN BIOLOGIE.

A. Variabilité biologique et groupes témoins.

Les résultats apparamment imprévisibles en biologie sont dûs à la variabilité importante observée.

Il est ,par exemple, difficile d'obtenir deux résultats identiques en mesurant le taux d'une substance à faible concentration dans le sang, d'étudier l'activité d'une enzyme sujette à de nombreux facteurs antagonistes et incontrôlables,...etc.

Il est toutefois possible de découvrir certaines règles dans cette variabilité. Tout n'est pas le fruit du hasard !

En ce qui concerne les groupes témoins (qui constituent la source des données que nous traitons dans le logiciel), il est bon d'en donner une signification et de parler de leur utilisation en biologie.

Vu la variabilité observée dans la nature, il est difficile d'établir, en fonction des résultats obtenus suite à une expérience, si la diversité des résultats est fonction du matériel utilisé ou bien dû à l'expérimentation elle-même.

C'est pourquoi on constitue, lors de toute expérience, un groupe (ici d'animaux) de témoins ne subissant aucun traitement particulier et servant de référence.
Ils sont mis exactement dans les mêmes conditions expérimentales que le groupe de test sur lequel on étudie l'effet d'une substance ou d'un traitement. Les mesures de résultats s'effectuent pour chacun des deux groupes.

Ceci permet de vérifier si un effet, mis en évidence dans le groupe test, n'est pas simplement dû à la variabilité naturelle existante chez les individus.
Il suffit pour cela de comparer le type de résultats obtenus pour le groupe témoin avec ceux du groupe test.

De plus, à partir des résultats obtenus avec le groupe de contrôle (groupe de témoins), il est possible d'établir une distribution d'échantillonnage permettant de dégager un intervalle de normalité pour ces valeurs témoins.

Cet intervalle permettra par la suite de vérifier si des résultats expérimentaux entrent ou non dans celui-ci, et donc d'en déduire le caractère normal ou anormal (v.chap.4.3.3).

B. Etude et signification statistique en biologie.

Si on peut considérer que le vrai et le faux doivent être remplacé par le probable et l'improbable dans ce domaine, les mécanismes de prise de décision et de connaissance des risques d'erreurs qui y sont liés, doivent être connus de tout expérimentateur.

C'est donc avec le plus grand intérêt qu'il utilise les outils statistiques offerts pour résoudre chaque type de problème.

La masse d'information récoltée et la nature des données font qu'il est indispensable de disposer de techniques statistiques permettant de dégager des tendances ou d'éprouver des hypothèses ,émises a priori, sur le type de matériel à tester.

Il doit aussi être possible d'établir, à partir d'un échantillon représentatif, les caractéristiques générales d'une population, de définir des valeurs ou des intervalles de normalité ou encore de tester l'effet (significatif ou non) d'une substance.

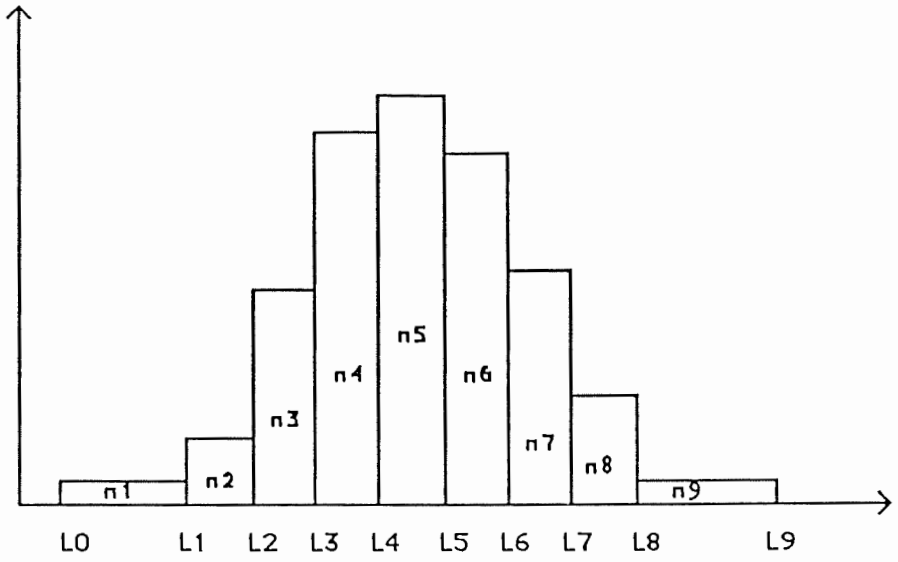
En ce qui concerne les tests d'hypothèses, il faut noter qu'il est important de connaître les risques d'erreur lié à toute prise de décision.

A côté du risque dit de première espèce (c'est-à-dire le risque d'accepter une hypothèse qui serait fausse), celui de deuxième espèce revêt une importance toute particulière en toxicologie ou en biologie clinique.

La probabilité de ne pas rejeter une hypothèse qui serait fausse doit être la plus faible possible.

En effet, il est, par exemple, dangereux de considérer qu'il n'y a pas d'effet toxique pour l'organisme mis en évidence pour une substance, alors que c'est le cas !

Il faut de plus garder à l'esprit que l'absence de signification statistique peut résulter plus de l'insuffisance du nombre de données que de l'absence réelle de différences au niveau des populations.



Legende :

Abscisse : Intervalles de classe (Li Li-1)

Effectifs de la classe i : ni

Hauteur d'un rectangle #ni / (Li -(Li - 1))

Effectif global : $n = \sum_{i} ni$

Aire d'un rectangle # ni

Figure 17 : Histogramme pour un échantillon de n individus.

La plus grande prudence s'impose donc dans l'analyse et la signification des résultats statistiques obtenus.

2. RAPPELS STATISTIQUES.

A. L'histogramme

Cette technique de représentation graphique de distributions de fréquence est utilisée en statistique descriptive. Nous l'utilisons dans la partie traitements statistiques offerts par notre logiciel.

Le principe en est le suivant. Quand le nombre de valeurs observées est élevé, il est souvent nécessaire de grouper les observations en classes ou catégories.

Ceci permet d'obtenir des distributions de fréquences groupées en classes; la fréquence d'une classe étant le nombre d'observations qui y sont contenues.

Ce type de distribution est particulièrement utile lorsqu'on dispose d'un grand nombre de données relatives à une variable continue dont les valeurs observées sont proches les unes des autres.

Elle peut être représentée graphiquement sous forme d'"histogramme" (v.fig.17).

L'histogramme est un diagramme d'aire . Pour le construire on procède comme suit: les limites de classes (définies par les valeurs extrêmes que ces classes peuvent contenir) sont portés en abscisse. Les intervalles de classe ne sont pas nécessairement égaux. Il est fréquent de regrouper les valeurs extrêmes dans des intervalles plus grands afin d'éviter des classes vides.

Sur base des limites de classe, on construit des rectangles dont la hauteur vaut :

$$\frac{\text{fréquence relative de la classe } (ni/n)}{\text{intervalle de la classe } (Li - (Li-1))}$$

L'histogramme se compose donc de rectangles contigus, dont les intervalles de classes sont les bases. Les aires des rectangles sont ainsi proportionnelles aux fréquences.

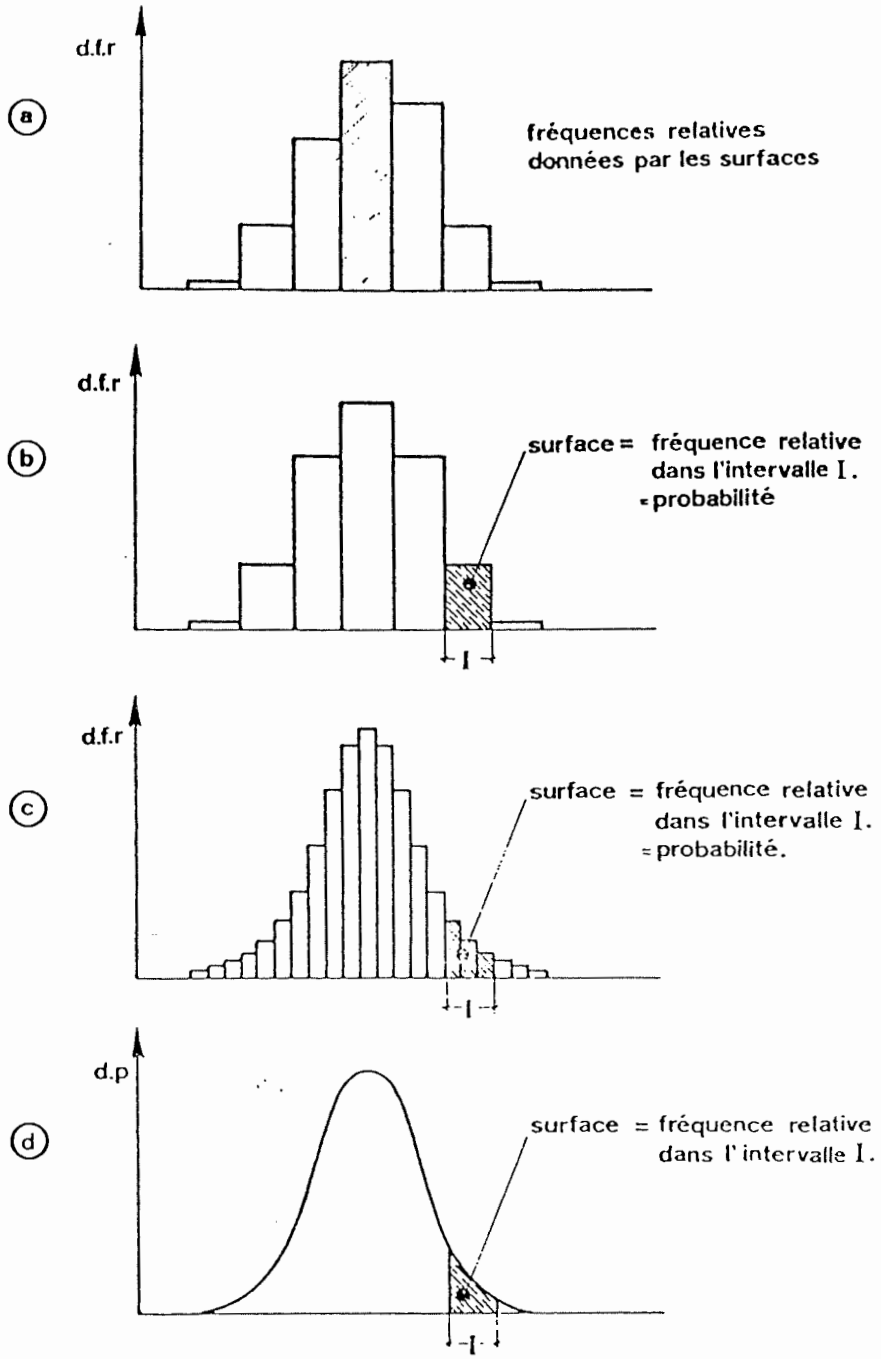


Figure 18: Effet produit par une augmentation de la taille de l'échantillon et par une diminution de l' intervalle de classe sur la distribution d' un échantillon avec en :

(a) un nombre n d'observations petit
(b) un nombre n d'observations grand
(c) un nombre n d'observations grand et des intervalles de classes plus petits
(d) le nombre n tend vers l'infini et l'intervalle de classe tend vers 0

(FEYTMANS 76 fig 4.6)

En utilisant la notion de densité de fréquence relative (c'est à dire $\frac{\text{FREQUENCE RELATIVE}}{\text{INTERVALLE DE CLASSE}}$) , il est possible de voir le lien qui existe entre les histogrammes et les distributions de probabilité [FEYTMANS 76].

La figure 18 illustre l'effet produit par une augmentation de la taille de l' échantillon et par une diminution de l' intervalle de classe . Avec un petit échantillon , les fluctuations dues au hasard influencent les résultats. Lorsque la taille de l'échantillon augmente , les fréquences relatives deviennent des probabilités , et les densités de fréquence relatives tendent vers des valeurs constantes , appelées densités de probabilité. En même temps, l'augmentation de la taille de l'échantillon permet de subdiviser plus finement les classes . La surface totale reste toujours égale à 1 , mais la densité de fréquences relatives devient approximativement une courbe , appelée fonction de densité de probabilité ou distribution de probabilité (f(X)).

Plus exactemement , la densité de probabilité f(x) est par définition la dérivée de la fonction de répartitition Pr [x $\leqslant$ a] , c'est-à-dire la probabilité que x soit inférieure ou égale à la valeur a .

B. Valeurs typiques

Certaines valeurs typiques (paramètres) permettent de caractériser les distributions observées.

Parmi celles les plus couramment utilisées, retenons :

- la moyenne : c'est la valeur moyenne observée (Σ Xi/n)
- la médiane : c'est la valeur telle que la moitié des observations lui sont inférieures et l'autre moitié, supérieures
- le mode : c'est la (les) classe(s) de fréquence maximum.

Outre ces paramètres de position servant à caractériser l'ordre de grandeur des observations, des paramètres de dispersion servent à chiffrer la variabilité des valeurs observées autour d'un paramètre de position.

Ceux dont nous nous servons sont:

- la variance : c'est la moyenne des carrés des écarts par rapport à la moyenne.
- l'écart type (déviation standard) : c'est la racine carrée de la variance.

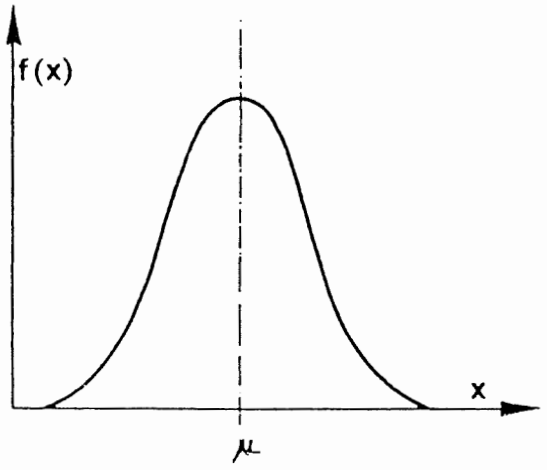


Figure 19 : Forme typique d'une distribution normale (FEYTMANS 76 fig 4.7)

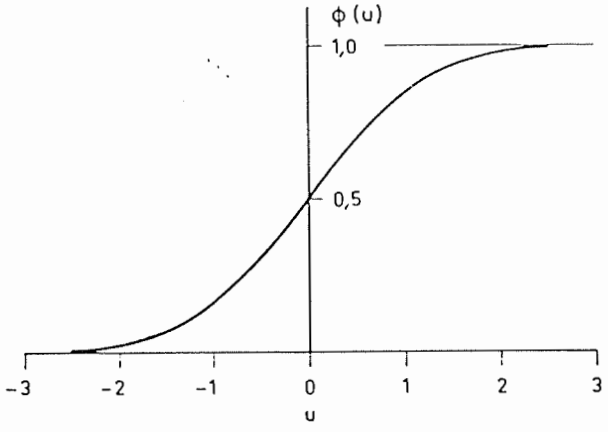


Figure 20 : Distribution normale réduite : fonction de répartition (DAGNELIE 73 p 220)

- le coefficient de variation : c'est l'expression de l'écart type par rapport à la moyenne; multiplié par cent, il fournit un pourcentage.

C. La distibution normale

Pour beaucoup de variables aléatoires , la distribution de probabilité est une courbe typique , en forme de cloche , appelée courbe normale ou courbe de Laplace-Gauss (figure 19).

C'est la distribution la plus utile en statistique ; par exemple , les erreurs commises sur des mesures de phénomènes physiques , biologiques , ou économiques , sont souvent distribuées normalement. De plus , certaines distributions , comme la binomiale , peuvent parfois être approximées par une courbe normale.

Si une variable aléatoire X est distribuée normalement , sa fonction de densité de probabilité est

$$f(X) = \frac{1}{\sigma \sqrt{2\pi}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

où les paramètres μ et σ^2 sont respectivement la moyenne et la variance de cette variable aléatoire X [PEYTMANS 76].

On dira que X est $N(\mu ; \sigma^2)$.

La fonction de répartition est l'intégrale de la fonction de densité de probabilité. Elle est représentée à la figure 20 pour une distribution normale réduite ($\mu = 0$, $\sigma^2 = 1$). Cette dernière est tabulée et sert de point de référence lors des tests d'hypothèses.

Le gros intérêt d'avoir une distribution normale est qu'il suffit de connaître la moyenne et la variance pour caractériser la distribution.

Il faut cependant remarquer qu'en pratique , seules la moyenne et la variance de l'échantillon , et non pas de la population , sont connues.

D. Tests de normalité

* But

voir si on a à faire à une distribution normale

* Test de Chi-carré

L'idée de base qui se trouve derrière le test de χ^2 est de comparer la distribution d'un échantillon Y avec la distribution attendue si l'échantillon était obtenu d'une population normale avec une moyenne égale à la moyenne de l'échantillon et une variance égale à la variance de l'échantillon.

Pour comparer la distribution de l'échantillon avec la distribution attendue , les observations sont d'abord triées et ensuite divisées en groupes. Les limites des groupes sont choisies de manière non aléatoires. Une fois les observations divisées en groupes , le nombre observé dans chaque groupe , Nobs , est compté et comparé avec le nombre attendu dans chaque groupe , Natt.

Le test χ^2 est calculé de la sorte :

$$\chi^2_{obs} = \sum_{i=1}^{p} \left(\frac{Nobs - Natt}{Natt} \right)^2$$

avec p le nombre de groupes.

Dans le cas où l'échantillon a une distribution normale , le χ^2_{obs} a une distribution de χ^2 .

La valeur χ^2_{obs} est nulle lorsque les fréquences observées sont toutes égales aux fréquences attendues , c'est à dire lorsqu' il y a concordance absolue entre la distribution observée et la distribution théorique.
D'autre part , cette valeur est d'autant plus grande que les écarts entre les fréquences observées et attendues sont plus grands. On rejettera donc l'hypothèse nulle lorsque la valeur observée est trop grande , c'est à dire lorsque :

$$\chi^2_{obs} \geq \chi^2_{1-\alpha}$$

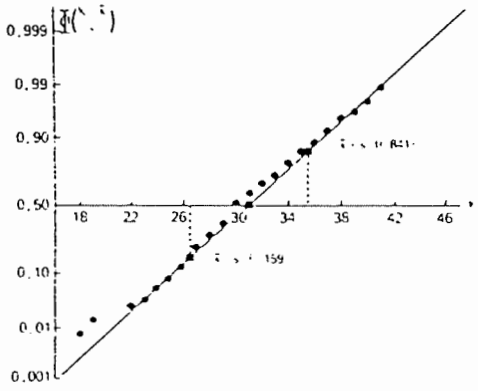


Figure 21: Test de normalité sur un échantillon de 143 valeurs par la procédure graphique (ou diagramme PROBIT) (ALBERT 83 fig 7)

avec α désignant la probabilité de rejet (vaut souvent 5 %).

Le $\chi^2_{1-\alpha}$ est obtenu à partir de tables préétablies [BBN 88].

* Test de Wilk-Shapiro

L'objectif de ce test de normalité est de déterminer si un échantillon aléatoire Y pourrait avoir été tiré d'une population normalement distribuée avec une moyenne μ inconnue et une variance σ^2.

Un rapport W est calculé : $W = \frac{(a^t y)^2}{s^2}$

où $a^t y$ est , à une constante près , le meilleur estimateur linéaire de la pente de la régression des observations ordonnées de l'échantillon Y , et s^2 est la variance de l'échantillon.

Pour un échantillon issu d'une population normale , le numérateur et le dénominateur du rapport W converge vers la variance σ^2 de la population.

Pour des populations non normales , ces quantités ne convergent pas vers la même quantité , le numérateur ayant tendance à être moindre que le dénominateur , rendant le rapport $W < 1$.

Il existe des tables de distribution de W qui permettent de rejeter ou d'accepter l'hypothèse de normalité [BBN 88].

* Discussion

Le test de Wilk-Shapiro est souvent utilisé pour des échantillons de petite taille (dans RS/1, c'est-à-dire le logiciel de statistiques utilisé (cfr.point 3.4), il est utilisé pour des échantillons de 50 observations).

D'autre part , le test de χ^2 est connu pour très bien fonctionner pour de grands échantillons (plus de 50 observations dans RS/1).

Il existe d'autres tests de normalité. Par exemple , la procédure graphique (ou "droite de Henry") et le test de Kolmogorov et Smirnov. La première de ces deux méthodes est illustrée à la figure 21 . Ce diagramme utilise un système d'axes dit anamorphose gaussienne : l'axe des ordonnées est gradué non linéairement de telle sorte que toute fonction de répartition normale est représentée par une droite. Un écart significatif à la droite témoigne en faveur d'une non normalité de l'échantillon [BBN 88].

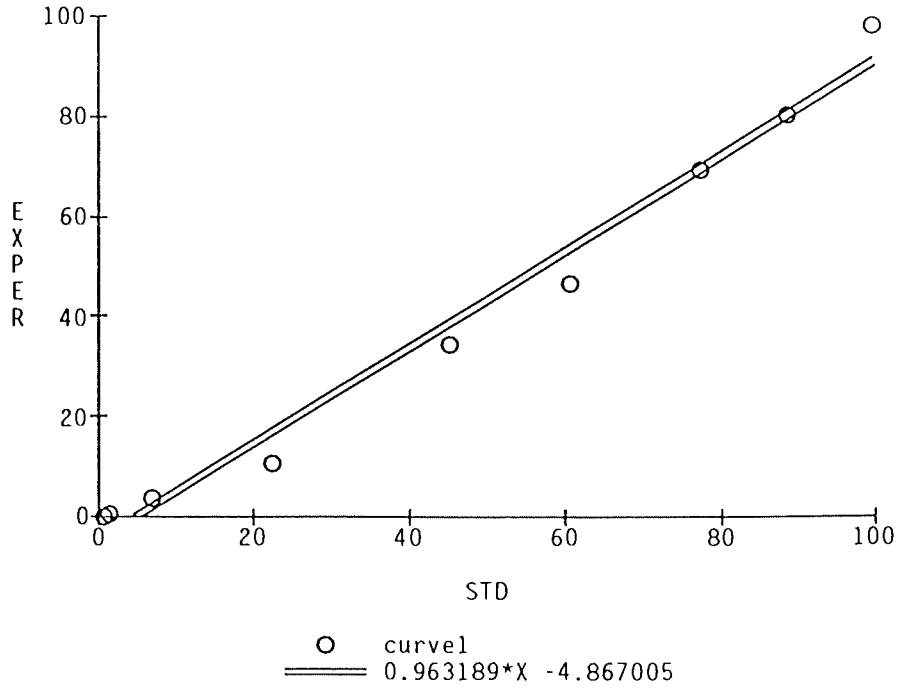


Figure 22 : Droite de regression.
(BBN 88 , p.5-4)

E. La regression

Cette technique a essentiellement pour but de mettre en évidence les relations existant entre deux séries d'observations considérées simultanément.
Ces observations peuvent être de nature qualitative ou quantitative, continue ou discontinue [FEYTMANS 86].

La droite de régression reflète l'allure, supposée linéaire, de la relation existant entre les deux séries de résultats.

Nous utiliserons souvent la régression dans la partie statistique de ce travail. C'est pourquoi il semble utile d'en rappeler les grands principes.

1° Conditions d'application

Dans le cas, auquel nous nous limiterons, où une des deux variables étudiées ne prends que certaines valeurs choisies a priori et de façon arbitraire, on suppose que :

- la variable indépendante X (variable dite explicative) n'est pas aléatoire. Elle est fixée par l'expérimentateur.

- la variable dépendante Y (variable à expliquer) est aléatoire.

2° La droite de régression

Outre une représentation graphique des observations, il faut pouvoir calculer les paramètres caractérisant numériquement les relations existant entre les deux variables.

Ainsi, lorsque le diagramme de régression de Y en fonction de X (formé par la représentation sur un système d'axes des points (xi,yi)) est linéaire ou approximativement linéaire, on peut s'efforcer de rechercher l'équation de la droite qui s'y ajuste le mieux (cfr.fig.22).

Cette droite de régression, dite aussi droite de régression de Y en X, est généralement déterminée par la méthode des moindres carrés, ou au sens des moindres carrés.

C'est-à-dire de manière à rendre minimum la somme des carrés des écarts entre les points observés et les points correspondants de la droite :

$$\text{Minimiser} \quad \sum_i (Yi - \hat{Yi})^2$$

Sachant que, d'une façon générale, tous les modèles linéaires peuvent s'écrire :

$$Y = Ab + E$$

avec Y : un vecteur d'observations
A : une matrice de coefficients
b : un vecteur de paramètres
E : un vecteur d'erreurs normales, indépendantes de moyenne nulle et de variance

Minimiser les erreurs revient donc à :

$$\text{Minimiser} \quad \sum_i (Yi - (Ab)i)^2$$

Afin d'estimer les paramètres, il convient d'en définir des estimateurs dont on connait la distribution. Par développements, on trouve :

$$\hat{b} = (A^T A)^{-1} A^T Y \qquad \text{avec } \hat{b} \text{ : estimateur de b}$$

et

$$\hat{Y} = A\hat{b} + \hat{E} \quad , \qquad \text{l'équation de la droite estimée.}$$

3° Cas particulier : la régression pondérée.

En effectuant la détermination des équations de régression à l'aide de la formule ci-dessus, on accorde implicitement une même importance à chacun des points observés (Xi,Yi).

Cette façon de faire ne se justifie que si ces points sont tous connus avec une même précision ou avec des précisions équivalentes.

Dans le cas contraire, il faut attribuer aux différentes valeurs observées des coefficients de pondération ou des poids différents (notés Wi).
On obtient alors une équation de régression pondérée.

Normalement les poids Wi sont choisis de manière inversément proportionnelle aux variances des Yi.

Par exemple, en supposant que la variance des Y est proportionnelle aux valeurs observées Yi (c'est-à-dire un coefficient de variation constant) les poids sont tels que :

$$Wi = 1 / Yi$$

A titre d'illustration, considérons le problème suivant. Soit, lors de pipettages de volume dans un laboratoire, un taux constant d'erreur de 1 %, quelque soit la pipette utilisée. Pour une pipette de 0,1 ml de capacité, le pourcentage d'erreur donne 0,1 * 1/100 = 0,001 ml en moyenne. Tandis que pour une pipette de 1 ml de capacité, on trouve 1 * 1/100 = 0,01 ml en moyenne. Soit dix fois plus d'erreurs qu'avec l'autre pipette !

Il est donc intéressant de donner une importance relative plus grande aux résultats obtenus, dans une même expérience, par le pipettage réalisé avec des pipettes de 0,1 ml.

On pondèrera les points de façon inversément proportionnelle à la variance observée.

4° Intervalle de confiance

La théorie de la régression permet l'estimation ou la prévision des valeurs de la variable dépendante en fonction de valeurs de la variable indépendante.

En effet, en posant le modèle $Y = aX + b$, on estime, sur base de l'échantillon, le a et le b. Ceci permet de trouver l'équation $\hat{Y} = \hat{a}X + \hat{b}$ de la droite estimée. Il est alors possible, pour un x donné, de prédire la valeur de Y correspondante.

Un problème annexe est la recherche des limites de confiance correspondantes. On désirera savoir entre quelle et quelle limite se situe normalement une valeur prédite.

Pour cela, on choisit arbitrairement un certain nombre de valeurs de X (Xi) ne correspondant pas nécessairement à des valeurs observées, et pour chacune d'elles, on estime les deux limites de confiance entre lesquelles se trouve le $aXi + b$.

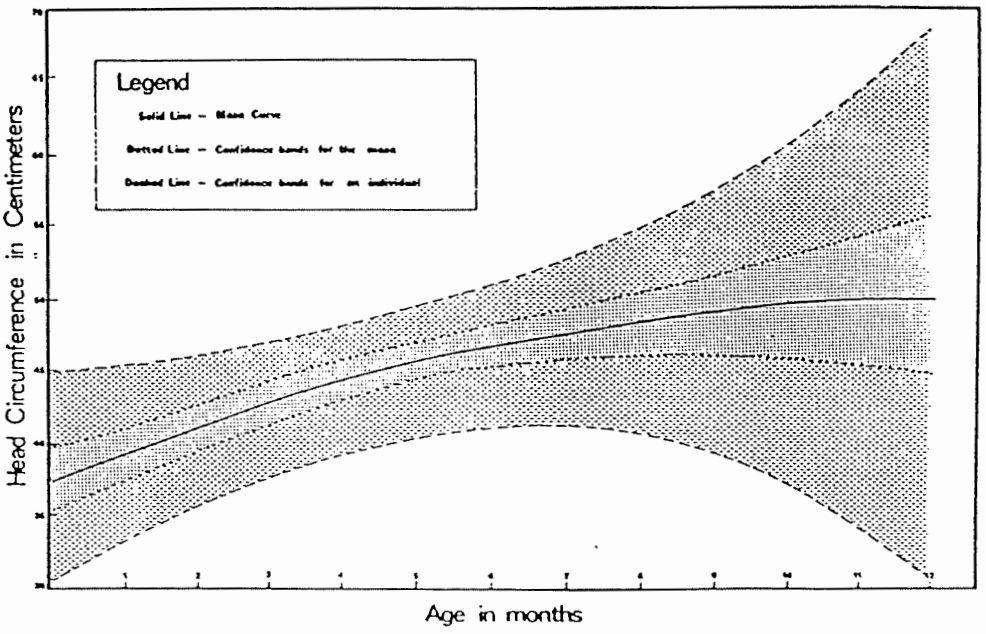


Figure 23: Intervalles de confiance d'une courbe de croissance d'une population estimée. (DAWSON 80, fig.3).

On obtient graphiquement une zone de confiance limitée par deux branches d'hyperbole qui s'écartent de plus en plus l'une de l'autre, de part et d'autre du point moyen (x,y). Ceci s'explique par le fait qu'on possède plus d'information pour ces valeurs centrales (cfr.fig.23).

5° La régression curvilinéaire

Utilisant cette technique dans la partie consacrée aux traitements statistiques (cfr.chap.4), il paraît intéressant de décrire brièvement celle-ci et d'en expliquer l'intérêt.

Dans de nombreux problèmes, une relation nette apparaît entre les variables étudiées sans que cette relation soit linéaire.

S'il est impossible de linéariser cette relation par des transformations mathématiques des données, il faut recourir à d'autres techniques.

Il peut ainsi être utile de procéder à l'ajustement (fitting) d'une courbe de régression au nuage des points observés. Dans certaines conditions, des tests de linéarité permettent d'ailleurs de contrôler d'une manière rigoureuse si l'ajustement d'une droite de régression est satisfaisant ou non.

Mais, lorsqu'il faut ajuster une courbe, deux problèmes se posent. Le choix de l'équation de cette courbe (c'est-à-dire d'un certain type de fonction mathématique) et la détermination des paramètres intervenant dans cette équation.

En ce qui concerne le choix d'une équation de régression, on peut se baser :

* soit sur des considérations théoriques, ce qui est de loin préférable car de cette façon, les équations obtenues s'adaptent mieux à la réalité du problème étudié et les paramètres possèdent souvent une signification précise;

* soit sur des considérations empiriques lorsqu'on ne dispose d'aucune base théorique satisfaisante. On tente alors de rechercher le type d'équation à ajuster en fonction des données observées elles-mêmes.

On s'efforce de transformer empiriquement ces données de manière à rendre linéaire la régression, ou bien on essaie d'ajuster empiriquement un polynôme de degré k :

$$Y = b_0 + b_1 x + b_2 x^2 + \ldots + b_k x^k \ ;$$

Ceci peut être également réalisé par la méthode des moindres carrés . L'équation générale à considérer est toujours Y = Ab + E, seul le vecteur paramètre b à utiliser diffère par rapport à la simple régression.

Cette méthode particulière de régression, appelée régression polynômiale ou parabolique, est d'un usage particulièrement courant, en raison surtout de sa grande facilité d'emploi.

Il est en effet toujours possible d'ajuster de façon satisfaisante un polynôme ou une parabole de degré k à un ensemble de points observés.

Cependant, il faut garder à l'esprit qu'il s'agit ici "d'un procédé d'ajustement purement mécanique qui conduit généralement à l'obtention d'une fonction dont les paramètres ne possèdent aucune signification propre. La régression polynomiale peut néanmoins donner une description satisfaisante du phénomène observé. Elle permet éventuellement d'en étudier certaines particularités, telles que les extremums, mais elle n'en fournit habituellement aucune interprétation valable" [DAGNELIE 73].

F. L'analyse de la variance

1. But et principe :

L'analyse de la variance à un critère de classification a pour but la comparaison des moyennes de populations, à partir d'échantillons aléatoires et indépendants, prélevés dans chacune d'elles.

Ces populations comparées sont généralement des variantes d'un facteur contrôlé de variation.

Ainsi, un biologiste souhaitant déterminer l'influence des régimes alimentaires sur le taux de glucose dans le sang de rat, compare les moyennes des taux chez les rats soumis respectivement à des régimes R1, R2, R3 ; dans une telle expérience, les trois populations comparées sont définies par les trois types de régime.

Dans le cas de l'analyse à un critère de classification (qui nous intéressera dans la partie

traitements statistiques du logiciel), les conditions qui doivent être remplies par les populations étudiées et par les échantillons prélevés, et qui permettent l'utilisation des méthodes d'inférence statistique, sont :

- la caractéristique numérique étudiée doit suivre une distribution normale
- les variances des diverses populations doivent être égales
- les échantillons doivent être prélevés de façon aléatoire et indépendante dans chacune des populations.

Le principe qui est à la base de l'analyse de la variance est [FEYTMANS 86]:

l'égalité des moyennes Mi (moyennes théoriques des populations) sera mise en doute si la dispersion des moyennes expérimentales X est grande comparativement à la dispersion des observations de chaque échantillon.

2. Le test d'hypothèse :

Pour tester l'hypothèse de l'égalité des moyennes de p populations

$$H : \mu_1 = \mu_2 = \ldots = \mu_p ,$$

on doit d'abord prélever un echantillon aléatoire et simple dans chaque population. Les moyennes de ces p échantillons et la moyenne générale de l'ensemble des observations permettent ensuite de définir deux types de variation.

- les écarts existant entre les différents échantillons (variation entre échantillons ou variation due au facteur contrôlé ou variation factorielle).
- les écarts existant à l'intérieur des échantillons (variation dans les échantillons ou variation résiduelle).

Le test d'égalité des moyennes revient à comparer le modèle

$$Y = Ab \qquad \text{avec} \qquad b = \begin{vmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_p \end{vmatrix} ,$$

c'est-à-dire le modèle de p populations normales, avec celui où toutes les observations sont extraites d'une même population de moyenne μ :

$$\mathcal{E}Y = Ab \qquad \text{avec} \qquad b = \mu \quad ,$$

c'est-à-dire le modèle d'une seule population normale.

Ceci revient à tester un modèle M2 par rapport à un modèle M1. Il s'agit d'un problème d'épreuve d'hypothèse que l'on traite en utilisant une méthode classique [NOIRHOMME 89].

On définit le vecteur v tel que

$$V = A_1 \hat{b}_1 - A_2 \hat{b}_2 = \hat{E}_2 - \hat{E}_1 \qquad (\text{avec } \hat{b}_1 : \text{estimateur de b})$$

est significatif de l'écart entre les deux modèles.
On définit aussi la région critique :

$$Rc = \{ W : r(w) > Q_{F(p-q;n-p)}(1-\alpha) \}$$

avec α = erreur de première espèce
p = le rang de A
q = le rang de A
$r = \dfrac{SCE_2 - SCE_1 \,/\, rgA_1 - rgA_2}{SCE_1 \,/\, n - rgA_1}$

(variable F de Snedecor à $rgA_1 - rgA_2$ et $n - rgA_1$ degrés de liberté)

L'hypothèse M2 sera rejetée par rapport à M1, au niveau , si l'observation a fourni un r(w) supérieur au quantile $Q_{F(p-q;n-p)}(1-\alpha)$!

3. Les modèles :

Suivant le choix des populations, on distingue deux types de problèmes différents ou "modèles"[FEYTMANS 86].

Le modèle fixe : les populations expérimentales résultent d'un choix non aléatoire; elles correspondent souvent à différents niveaux ou variantes d'un traitement ou facteur contrôlé qui est appelé fixe.

Ex. : - comparaison des gains en poids d'animaux soumis à différents régimes alimentaires.

- comparaison du métabolisme de différentes souches de souris.

Le modèle aléatoire : les populations étudiées constituent un échantillon, prélevé de façon aléatoire, dans un ensemble plus large de populations. Dans ce cas, le facteur contrôlé est appelé "aléatoire".

Ex.: détermination de la teneur en glucose d'un organe particulier.

Bien que l'analyse globale des résultats expérimentaux et le test de comparaison se réalisent de la même façon, il ne faut pas confondre ces deux modèles.

Le modèle fixe est plus restrictif: les conclusions déduites ne concernent que les populations qui ont été effectivement étudiées.

Par contre, pour le modèle aléatoire, les conclusions peuvent être étendues à l'ensemble des populations desquelles seulement un échantillon aléatoire a été étudié.

5. Comparaisons multiples des moyennes

Lorsque l'analyse de la variance a conduit au rejet de l'égalité des moyennes, on peut se demander quelles sont, parmi les p populations, celles qui diffèrent significativement entre elles.

Les méthodes de comparaisons multiples ont pour but de résoudre ce problème. En fait, celui-ci peut se présenter sous différents aspects auxquels correspondent différentes méthodes d'analyse.

En ce qui concerne le cas particulier (qui nous intéressera ici) de comparaison de p moyennes deux à deux, différentes techniques sont proposées.

La première, souvent appelée méthode de la plus petite différence significative, consiste à effectuer des comparaisons de moyennes deux à deux grâce au test t de Student ($t = \frac{\overline{X} - \overline{Y}}{\sqrt{\frac{s_x^2}{n_x} + \frac{s_y^2}{n_y}}}$) avec $\overline{X}$, $\overline{Y}$: moyennes des échantillons
s_x , s_y : variances des échantillons
n_x , n_y : nombre d'observations.

Du fait de la multiplicité des tests d'une part et de la non indépendance des comparaisons effectuées d'autre part, le risque global d'erreur de première espèce (c'est-à-dire la probabilité de mettre en évidence au moins une différence significative alors que les moyennes de toutes les populations sont égales entre elles) peut être très important.

Pour remédier à cela, on peut soit diminuer le niveau de signification afin de réduire le risque d'erreur globale (mais c'est une procédure arbitraire), soit utiliser des méthodes plus objectives.

Parmi celles-ci, nous retiendrons la méthode de Newman et Keuls (basée sur la comparaison d'amplitudes observées pour des groupes de p moyennes, avec l'amplitude maximum attendue à un niveau de signification donné) et la méthode de Duncan (qui utilise des valeurs critiques différentes de celles préconisées par Newman et Keuls).

Les tests de Duncan et de Newman-Keuls utilisent le même algorithme:

étant donné k groupes avec N_k observations dans chaque groupe et des moyennes d'échantillon $\overline{X}_1$, $\overline{X}_2$, ... $\overline{X}_k$, il y a un total de $n = \sum N_k$ et on suppose que, pour une question de simplicité, $\overline{X}_1 < \overline{X}_2 < \ldots < \overline{X}_k$.

1. on choisit d'abord la plus grande moyenne d'un groupe, $\overline{X}_k$, et la plus petite, $\overline{X}_1$. On calcule ensuite la statistique :

$$q^{\alpha_k}_{k,(m-k)} = \frac{\overline{X}_k - \overline{X}_1}{s\sqrt{\frac{1}{N_k} + \frac{1}{N_1}}}$$

La statistique $q^{\alpha_k}_{k,(m-k)}$ est la statistique de la distibution de Student. La quantité s au dénominateur est la racine carrée du carré moyen de l'erreur résiduelle provenant de l'analyse de la variance.

2. on compare la statistique $q^{\alpha_k}_{k,(m-k)}$ avec une valeur critique pour un $\alpha = 0.05$ (test bilatéral) tiré d'une table. Si la valeur calculée dépasse la valeur critique, le test est significatif.

3. si le test est significatif, on répète l'opération en comparant la plus grande moyenne avec la deuxième plus petite moyenne.

La procedure se poursuit de cette manière, testant la moyenne la plus grande avec chacune des autres moyennes.

Si à un moment donné, la moyenne la plus importante n'est pas significativement différente de la moyenne d'un autre groupe, plus aucun test n'est effectué avec la plus grande moyenne.

Une fois toutes les comparaisons effectuées avec la plus grande moyenne, le cycle recommence en testant la deuxième moyenne par ordre d'importance $\overline{X}_{k-1}$ avec la plus petite $\overline{X}_1$ et ainsi de suite.

La différence existant entre le test de Duncan et le test de Newman-Keuls réside entièrement dans la sélection des valeurs critiques de la statistique . Le test de Student de Newman-Keuls utilise $\alpha_k = 0.05$ alors que le test de Duncan utilise $\alpha_k = (1-(1-0.05)^k)$.

Pour des comparaisons impliquant plus de deux moyennes, le test de Duncan a un niveau de signification supérieur, ou en d'autres mots, une probabilité supérieure de rejeter l'hypothèse nulle selon laquelle toutes les moyennes sont égales alors qu'elle est fausse .

La raison avancée pour l'utilisation du test de Duncan est qu'il a une plus grande puissance que le test de Newman-Keuls. Cette raison n'est pas bien acceptée dans la littérature à cause de l'accroissement de l'erreur du premier type.

La validité des valeurs critiques utilisées dans chacun des tests dépend du fait que les groupes proviennent d'échantillons indépendants tirés de populations dont les variances sont égales. Si la taille des groupes varie, ou si les variances pour les groupes varient, ces tests doivent être interprètés avec précaution.

Pour plus de détails, on peut se référer aux ouvrages traitant du sujet ([DAGNELIE 73],[BBN 88]).

G. Méthodes de détermination des valeurs de référence

Pour rappel , le quantile α (ou d'ordre α) de la distribution d'un constituant biologique X est la concentration (ou l'activité) en deça de laquelle la probabilité de trouver une valeur égale α . C'est donc la valeur X_α telle que $Pr(X \leq X_\alpha) = \alpha$. Par exemple , pour une distribution gaussienne , c'est-à-dire normale de moyenne 0 et d'écart type 1 , le quantile 0.975 est égal à 1.96 car $Pr(X \leq 1.96) = 0.975$. De même le quantile 0.025 est égal à -1.96 .

a) Méthode non paramétrique

Nous décrivons la méthode non paramétrique d'estimation des quantiles. Soient X1,X2,...,Xn les valeurs de référence observées. Ordonnons-les de manière croissante , ce que l'on écrit :

$$X(1) \leq X(2) \leq \ldots \leq X(n-1) \leq X(n)$$

L'indice entre parenthèses donne le rang de l'observation. Donc X(1) est la plus petite valeur , X(2) la seconde plus petite valeur et ainsi de suite : X(n) est la plus grande valeur. Des valeurs égales ont forcément des rangs différents.

Estimation des quantiles : pour estimer le quantile $X_{0.025}$, il suffit de prendre la valeur dont le rang est égal à 0.025 * (n + 1) . De même , pour estimer le quantile $X_{0.975}$, on recherche la valeur dont le rang est égal à 0.975 * (n + 1) . Il arrive souvent que les nombres ainsi obtenus ne soient pas des entiers ; dans ce cas , on interpole le quantile entre les deux rangs qui contiennent ce nombre.

Supposons que n = 143 , on obtient 0.025 * 144 =3.6. Le quantile $X_{0.025}$ est obtenu en interpolant entre les valeurs des rangs 3 et 4 puisque 3 < 3.6 < 4 . De même , pour $X_{0.975}$.

La recherche des quantiles est effectuée pour obtenir 95 % de la population de référence. Le calcul pour obtenir les quantiles $X_{0.995}$ et $X_{0.005}$ en couvrant 99 % de la population de référence peut être réalisé de la même manière. Ceci a pour effet de diminuer la sensibilité du test et d'augmenter sa spécificité [ALBERT 83],[SOLBERG 87b].

b) Méthode paramétrique

Elle est applicable à des distributions normales.

Estimation des quantiles : on calcule la moyenne arithmétique $\bar{X}$ et l'écart type s de l'échantillon des n valeurs de référence. Les quantiles estimés sont alors donnés par les deux relations (avec un seuil de 5 %) :

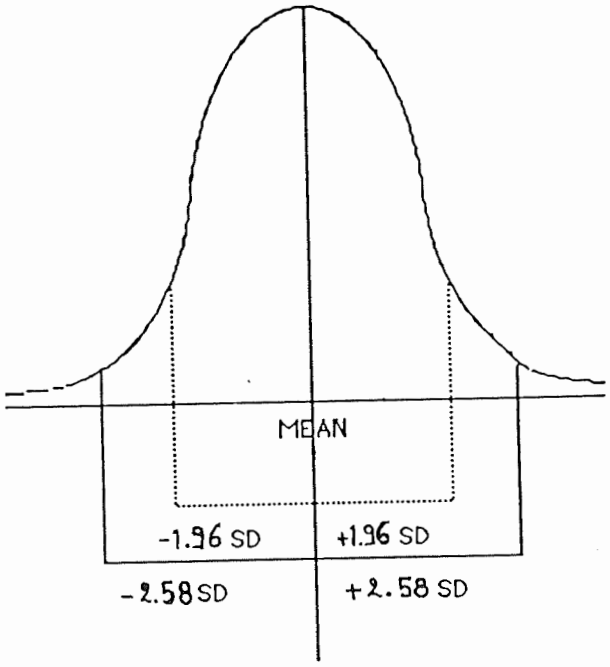
$$X_{0.025} = \bar{X} - 1.96 \; s$$

$$X_{0.975} = \bar{X} + 1.96 \; s$$

La valeur 1.96 est le quantile 0.975 de la distribution normale de moyenne 0 et d'écart type 1. Le fait que l'on ait 1.96 de chaque côté provient du caractère symétrique de la distribution normale [ALBERT 83], [SOLBERG 87b].

La remarque donnée dans le paragraphe a) pour un seuil de 1 % est également valable ici.

Voici la signification graphique de la méthode à l'instant décrite :



3.4. LE LOGICIEL RS/1 : PROGRAMMATION EN RPL.

Le logiciel RS/1 est un système d'analyse de données spécialement conçu pour satisfaire les besoins des scientifiques pour les traitements de données qu'ils ont à effectuer.

La structure de base dans RS/1 est la "table". C'est là que les données doivent être stockées, celles-ci pouvant être de types fort différents. Une telle table rassemble des colonnes, des rangées (toutes deux ayant leur numérotation commençant par zéro), des titres, des entêtes de colonne et des noms de rangée.

Les opérations offertes sur ces tables sont :

+ la création : MAKE TABLE nom-table FROM nom-table

+ la suppression : DELETE nom-table

+ l'ajout de nouvelles colonnes ou rangées :
  INSERT nom-colonne/nom-rangée

+ la suppression de colonnes ou rangées :
  DELETE col/row

A partir de ces tables, il est possible de réaliser des graphiques de statistiques. Des opérations analogues à celles données ci-dessus peuvent être effectuées sur les graphiques (créàtion, suppression de graphique,ajout, suppression de courbes,...etc.).

Nous nous proposons de détailler ici le langage de programmation RPL qui est intégré à l'environnement RS/1. C'est avec ce langage que nous avons élaboré nos modules de traitements statistiques (cfr."Conception du logiciel"). Après avoir succintement expliqué les structures de base du langage, nous expliquerons les routines de RS/1 utilisées. En effet, RS/1 offre une librairie de modules très spécifiques qui peuvent être appelés à partir d'une procédure créée avec RPL.

A. Structures de base du langage RPL.

Les opérations réalisées vont donc l'être sur des tables ou sur des graphiques. On retrouve dans RPL les structures de programmation classiques, à savoir :

+ l'affectation :
    - concernant une TABLE :
        SET case d'une table TO variable,
                                case d'une table,...etc.
    - concernant une VARIABLE :
        variable = variable

+ la condition : IF condition THEN action 1 ELSE action 2

+ la répétition conditionnelle : DO WHILE condition

+ la répétition non conditionnelle : DO valeur 1 TO valeur 2
+ le branchement : GOTO label

En plus de ces structures classiques, certaines particularités sont offertes :

+ DONEXT : passage à l'itération suivante

+ DOEXIT : interruption de la boucle

+ FOR EACH CELL : permet de considérer toutes les cellules d'une table une à une.

+ FOR EACH ROW : permet de considérer toutes les rangées d'une table une à une.

+ FOR EACH COL : permet de considérer toutes les colonnes d'une table une à une.

En outre, on a la possibilité de travailler sur des tableaux, des matrices et des vecteurs. Ces deux dernières structures ne pouvant comporter que des valeurs numériques.

De plus, des instructions classiques d'affichage (fonction DISPLAY) et de saisie de données (fonction GET) sont offertes.

B. Routines de RS/1 utilisées.

Parmi l'ensemble des routines disponibles dans RS/1 ([BBN 88]), nous décrivons ci-dessus les routines que nous utiliserons dans la partie traitement statistique du logiciel. Il est à noter que les arguments mis entre crochets dans la description d'une procédure RS/1 sont facultatifs. Ils sont donnés à la suite des arguments obligatoires.

1. FITLINE :

But : ajuster une fonction linéaire (du type aX+b) à un ensemble de points d'un graphe.

Syntaxe : $FITLINE(graph,curve_number [,fitted_function, add_curve_flag, anova_table, coefficients_table, residuals_table, interactive_flag])

graph : nom du gaphe contenant la courbe de données

curve_number : numéro de la courbe de données à ajuster
fitted_function : fichier contenant l'équation ajustée
add_curve_flag : permet d'ajouter (TRUE) on non (FALSE) la droite ajustée sur le graphe
anova_table : nom de la table contenant les données de l'analyse de la variance
coefficients_table : nom de la table contenant les coefficients ajustés
residuals_table : nom de la table contenant les résidus
interactive_flag : entrée des arguments interactive (TRUE) ou non (FALSE)

2. FITPOLYNOMIAL

But : ajuster une fonction polynômiale ($b_0 + b_1 x + \ldots + b_m x^m$) à un ensemble de points sur un graphe.

Syntaxe : $FITPOLYNOMIAL(graph,curve_number, degree_number [,fitted_function, add_curve_flag, anova_table, coefficients_table, residuals_table, interactive_flag])

graph : nom du gaphe contenant la courbe de données
curve_number : numéro de la courbe de données à ajuster
degree_number : degré du polynôme à ajuster
fitted_function : fichier contenant l'équation ajustée
add_curve_flag : permet d'ajouter (TRUE) on non (FALSE) la droite ajustée sur le graphe
anova_table : nom de la table contenant les données de l'analyse de la variance

```
coefficients_table : nom de la table contenant les
                     coefficients ajustés
residuals_table : nom de la table contenant les
                  résidus
interactive_flag : entrée des arguments
                   interactive (TRUE) ou non
                   (FALSE)
```

3. FITFUNCTION

But : ajuster une fonction choisie dont on a donné l'équation à un ensemble de points sur un graphe. Ceci n'est intéressant que lorsqu'on connait le type de distribution que les points expérimentaux devraient adopter. Il faut fournir des estimations des coefficients qui interviennent dans l'équation.

Syntaxe :
```
$FITFUNCTION(function_text,graph,curve_number,
             weight_table, weight_columns,
             maximum_iterations, convergence_criterion,
             initial_values_table, fitted_function [,
             add_curve_flag, summary_table,
             anova_table, parameters_table,
             residuals_table, interactive_flag,
             msg_flag, function_proc,
             comf_intervals_graph, prediction_function,
             add_ci_flag ])
```

```
function_text : fonction à ajuster
graph : nom du gaphe contenant la courbe de
        données
curve_number : numéro de la courbe de données à
               ajuster
weight_table : nom de la table contenant les poids
               pour l'ajustement (EMPTY pour un
               ajustement non pondéré )
weight_columns : numéro de la colonne de cette
                 table
maximum_iterations : nombre maximum d'itérations
convergence_criterion : critère de convergence
                        (défaut 0.0001)
initial_values_table : table contenant les valeurs
                       initiales pour les
                       paramètres à ajuster
fitted_function : fichier contenant l'équation
                  ajustée
add_curve_flag : permet d'ajouter (TRUE) on non
                 (FALSE) la droite ajustée sur le
                 graphe
summary_table : nom à donner à la table contenant
                le résumé des traitements
                effectués
```

anova_table : nom de la table contenant les
données de l'analyse de la variance
parameters_table : nom à donner à la table
contenant les paramètres
residuals_table : nom de la table contenant les
résidus
interactive_flag : entrée des arguments
interactive (TRUE) ou non
(FALSE)
msg_flag : affichage du résultat au terminal
(TRUE) ou non (FALSE)
function_proc : nom d'une ancienne fonction
d'évaluation.
conf_intervals_graph : nom du graphe contenant les
intervalles de confiance
prediction_function : nom de la fonction prédite
qui est retournée
add_ci_flag :ajoute (true) ou non (false) les
intervalles de confiance sur le
graphe.

4. EVALCURVE

But : donner le Y estimé en fonction d'une valeur de X pour
une équation déterminée

Syntaxe : $EVALCURVE([graph, curve_number, X-value])

graph : nom du graphe contenant la droite pour
laquelle on désire des estimations.
curve_number : numéro de la courbe à considérer
x_value : valeur de x pour laquelle on désire
estimer une valeur de Y

5. GRAPH_CI

But : ajouter les intervalles de confiance d'une droite de
régression.

Syntaxe : $_GRAPH_CI(graph, curve_number, alpha, msg_flag)

graph : nom du graphe
curve_number : numéro de la courbe
alpha : niveau de confiance (défaut : 0.95)
msg_flag : procédure interactive (true) ou non
(false)

6. TESTNORMAL

But : tester si un échantillon de données provient d'une distribution normale. Il utilise les méthodes décrites au chapitre 3.3. (Chi-carré, Wilk-Shapiro).

Syntaxe : $TESTNORMAL (data_sample, [, normality_statistic, significance_level, msg_flag, result_text])

data_sample : table ou portion d'une table de données
normality_statistic : retourne la valeur du test (Chi-carré pour plus de 50 observations, Wilk-shapiro pour entre 3 et 50 observations)

significance_level : retourne le niveau de signification
msg_flag : affichage des résultats (true) ou non (false)
result_text : texte retourné décrivant les résultats

7. HISTOGRAM

But : construire un histogramme

Syntaxe : $HISTOGRAM([data_tp, histogram_graph, autodisplay, low_bound, high_bound, num_intervals])

data_tp : portion de la table contenant les données
histogram_graph : nom à donner au graphe obtenu
autodisplay : affichage du graphe (true) ou non (false)
low_bound : limite inférieure des intervalles
high_bound : limite supérieure des intervalles
num_intervals : nombre d'intervalles

8. ANOVAONEWAY

But : établir une analyse de la variance à un critère de classification.

Syntaxe : $ANOVAONEWAY (data_tp, factor_tp [,anova_table, msg_flag, mult_comp_graph, summary_table])

data_tp : portion de la table contenant les données
factor_tp : portion de la table contenant les catégories auxquelles les données sont rattachées

anova_table : table contenant les résultat de l'analyse de la variance
msg_flag : affichage des résultats (true) ou non (false)
mult_comp_graph : nom du graphe contenant les intervalles de confiance pour la moyenne de chaque groupe
summary_table : nom de la table contenant les résumés des statistiques

9. SIMULTANEOUS

But : établir des comparaisons multiples de moyennes. On localise ici l'effet mis en évidence lors de l'analyse de la variance ($ANOVAONEWAY).

Syntaxe : $SIMULTANEOUS(summary_table, test_type [, control_row, mean_squared_error, error_df, results_table, interactive_flag])

summary_table : nom de la table contenant les statistiques obtenues pour les différents groupes lors de l'analyse de la variance (appel de la procédure ANOVAONEWAY)
test_type : type de test à appliquer (Duncan ou Newman-keuls ou Dunnet)
control_row : - dans le cas du test de Dunnet, spécifie le nombre de rangées dans la "summary_table" qui contient des statistiques sur le groupe de contrôle
- dans les autres cas, valeur EMPTY
mean_squared_error : carré moyen des erreurs intra-groupe provenant de la table "SUMMARY_TABLE" si mis à EMPTY; ou bien il faut le spécifier
error_df : degré de liberté des erreurs de la table "Summary_table" ou bien spécifié à partir de la table de l'analyse de la variance
results_table : nom à donner pour la table des résultats retournés
interactive_flag : interactif (true) ou non (false)

10. MEASURE

But : calculer les statistiques descriptives à partir d'une table de données.

Syntaxe : $MEASURE(data_tp, results_table [,collapse_flag])

data_tp : table ou portion de la table contenant
les données.
results_table : nom à donner à la table contenant
les résultats de l'appel
collapse_flag : TRUE pour considérer l'entièreté de
la table, FALSE pour considérer
séparément chaque rangée ou colonne

11. PROBCHI

But : calculer la probabilité d'obtenir une valeur de la
distribution de Chi-carré.

Syntaxe : $PROBCHI(Chi_value,df)

Chi_value : valeur pour laquelle on veut calculer
la probabilité
df : degrés de liberté pour la valeur de Chi.

CHAPITRE 4 : C O N C E P T I O N D U L O G I C I E L

4. CONCEPTION DU LOGICIEL.

4.1 ENVIRONNEMENT GENERAL.

4.1.1 Matériel et logiciel.

Le matériel mis à notre disposition par Continental Pharma pour l'élaboration du logiciel consiste en un ordinateur VAX 8600 de la firme Digital, tournant sous le système d'exploitation VMS (Virtual memory system) version 4.7.

C'est sous ce matériel que nous avons développé notre logiciel. Celui-ci a été élaboré avec l'environnement de programmation Powerhouse (cfr.chap.3.2).

Pour la mise au point de certains modules d'interface entre notre base de données et les logiciels contenant les données et mesures à transférer, nous avons utilisé le langage de programmation Pascal (version 3.8 du logiciel Pascal standard installé sur Vax).

En outre, un package statistique a été choisi pour la conception des traitements statistiques. Il s'agit du logiciel RS/1 (cfr.chap.3.4) commercialisé par la firme BBN. Après une étude comparative de ce logiciel avec le logiciel de statistiques SAS, nous avons porté notre choix sur le premier essentiellement en raison du fait qu'il est capable de développer de meilleurs graphiques que SAS.

4.1.2 Environnement de développement.

Bien qu'il existe un cycle de développement classique bien reconnu, chaque entreprise adapte quelque peu celui-ci aux moyens dont elle dispose.

Les logiciels utilisés et le matériel informatique, sont des facteurs qui peuvent moduler le cycle. L'utilisation de Powerhouse en temps que langage de quatrième génération a un effet sur le cycle de vie comme il en a été débattu au point consacré aux langages de quatrième génération.

La taille du service informatique influence aussi la méthodologie de travail : il n'y a actuellement que trois analystes responsables du développement d'application. Chaque application à developper est confiée à une personne qui en réalise toutes les étapes. Ceci est rendu possible par le langage utilisé : Powerhouse.

Une première étape consiste à rédiger le "cahier des charges" qui est le résultat de la démarche de l'analyste vers le(s) utilisateur(s).
Les points essentiels demandés y sont dégagés suite à un certain nombre de réunions avec les intéressés.

Cette étape est délicate car bien souvent, il existe une certaine nébulosité du côté des utilisateurs quant à savoir ce qu'ils désirent exactement.

Le cahier ainsi rédigé est soumis par la suite aux personnes liées au projet en cours afin d'établir leur approbation et surtout leurs remarques.

Dans un deuxième temps, l'analyste étudie le problème de plus près et essaie de dégager les étapes fondamentales, le délai de réalisation du projet ainsi que le ou les langages qui vont être utilisés (v.rem).

Ceci débouche sur la rédaction du "rapport de faisabilité". Comme pour le "cahier des charges", il est soumis aux responsables du projet et aux utilisateurs qui fournissent leurs commentaires. Il consiste essentiellement à établir le caractère "faisable" de la réalisation du projet en fonction de ce dont on dispose.

Il est à noter que plusieurs versions de ces deux types de rapport peuvent avoir été rédigés en fonction de changements apportés suites à diverses précisions.

Ensuite, l'analyste conçoit toute l'application et entreprend le codage. Après un délai imparti au développement, le logiciel est mis en production.

Des modifications peuvent être apportées par la suite si nécessaire.

rem.: le langage Powerhouse est en principe utilisé mais il peut dans certains cas être nécessaire de se servir du langage de programmation Pascal. Ceci se fait via la création d'une librairie de modules appelables à partir de Powerhouse (plus précisément, à partir d'un écran réalisé en Powerhouse).

4.2 ETAPES D'UNE ETUDE DE TOXICOLOGIE.

4.2.1 Qu'est ce qu'une étude de toxicologie?

Une étude de toxicologie a pour but de mettre en évidence la toxicité éventuelle d'un médicament. Pour la tester , les animaux sont répartis en groupes qui reçoivent différentes doses du médicament. Pour que l'étude ait une signification , il est indispensable de se référer à un groupe témoin qui n'a reçu aucune administration du médicament. Les comparaisons se feront donc par rapport à ce dernier groupe.

Il y a trois grandes phases dans une étude de toxicologie précédées par l'entrée en quarantaine :

- la phase PRETEST
- la phase TEST
- la phase terminale

Il convient d'abord de parler de la quarantaine. Lorsque les animaux sont livrés par l'animalier , ils passent un certain temps (une semaine pour les rongeurs) en quarantaine. Durant cette période , un certain nombre d'observations sont effectuées ; il faut s'assurer que les animaux sont sains , éviter que les animaux trop malingres ou trop forts ne soient utilisés dans les études de toxicologie. Dans ce but , des mesures de poids , de consommation alimentaire , de paramètres biologiques sont réalisées. Donc, parmi les animaux livrés , un certain nombre seulement seront effectivement destinés à des études.

Après cette première sélection vient l'étude proprement dite :

1. phase PRETEST : les animaux ont été répartis pour chaque étude et cette première phase a pour but d'éliminer les animaux jugés anormaux par des observations plus limitées que celles qu'ils ont subi en quarantaine.

2. phase TEST : c'est à ce moment que sont formés les groupes. Ceux-ci se voient assignés tous les animaux jugés sains en n'oubliant pas le groupe contrôle (ou témoin).
Pendant cette phase , les différents groupes sont soumis à des doses variables du médicament et des mesures de poids (toxico) et de paramètres biologiques (biologie clinique) sont réalisées.

3. phase terminale : les deux phases précédentes sont souvent référées comme "in life". Dans cette dernière étape , des autopsies sont réalisées avec des mesures de poids d' organe , des observations des tissus. Cette phase nous intéresse moins pour le projet à réaliser.

Pour une étude , la numérotation des jours commence à la phase TEST. Ceci a pour conséquence que :

- les jours "supérieurs à 0" se réfèrent à la phase TEST
- les jours "inférieurs à 0" se réfèrent à la phase PRETEST

Une étude concerne une espèce et une race déterminées.

4.2.2 Numérotation des animaux

Celle-ci est assez complexe et change suivant que les animaux sont des rongeurs ou non et suivant les phases dans laquelle ils se trouvent. La tableau ci-dessous résume la nomenclature adoptée.

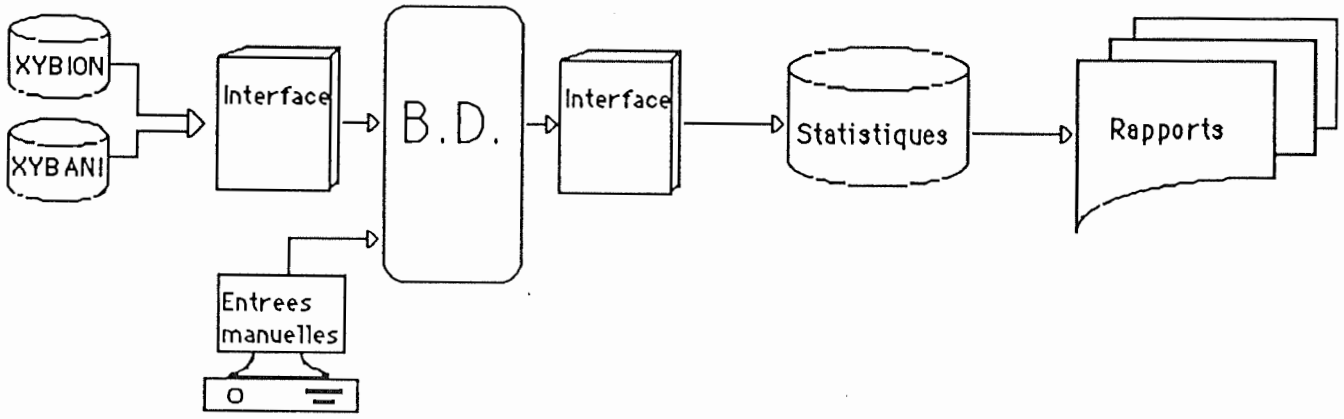
	QUARANTAINE	PRETEST	TEST
ROONGEURS	- Pas d'attribution de numéros individuels - Seulement un numéro de lot pour tous les animaux	Attribution de numéros individuels constitués par le numéro de lot + un numéro séquentiel	Attribution de nouveaux numéros individuels aux animaux restants constitués par le numero d' étude + un numéro séquentiel définitif
NON-ROONGEURS	Attribution de numéros individuels constitués par le numéro de lot à l'entrée + un numéro séquentiel	Pas de modification des numéros pour les animaux restants	Attribution de nouveaux numéros individuels aux animaux restants constitués par le numero d' étude + un numéro séquentiel définitif

4.3 ANALYSE DES BESOINS.

Comme il l'a été dit dans le but du travail , ce travail consiste à concevoir une "base de données" historique peuplée , entre autre , de paramètres biologiques. En réalité celle-ci n'est pas basée sur un Système de Gestion de Base de Données (SGBD) mais sur une manipulation de fichiers RMS ! La B.D ainsi constituée ne comportera que des valeurs de référence , les animaux pour lesquels ces valeurs ont été obtenues faisant partie des groupes de contrôle lors d'études toxicologiques ou étant issus de la quarantaine (ils ne font donc pas partie de telles études).

La manipulation de cette base de données permettra d'extraire des valeurs de ces groupes témoins afin d'essayer d'établir des distributions normales de ces paramètres , des régressions en fonction du poids , de l'âge , etc...

Avant de détailler les besoins , voici un schéma donnant une vue d'ensemble du problème :



Ce schéma général suggère deux parties dans le projet:

* la constitution de la B.D. ; comprenant le transfert des données , la conception de la B.D. et tous les problèmes liés aux conversions
  (point 4.3.1)

* l'exploitation de la B.D. ; comprenant la sélection des données et les traitements statistiques s'y rapportant
  (point 4.3.2)

4.3.1 Constitution de la B.D.

1. Provenance des données

Les données proviennent :

1) du logiciel XYBION : XYBION est un logiciel utilisé pour le stockage de données de biologie clinique pour des animaux en étude. Il constitue la source essentielle des données qui peuplent la B.D. et concerne toutes les études réalisées depuis quelques années.

Le prélèvement des données de XYBION se fera par une extraction selon le code d'une étude pour un déterminant (**) donné (c'est-à-dire pour un ou plusieurs paramètres).

Les informations obtenues seront :

- nom du déterminant,
- pour chaque paramètre :
  * nom du paramètre
  * numéro d'animal
  * sexe de l'animal
  * groupe
  * sous-groupe
  * jour de l'étude
  * valeur du résultat.

Le numéro d'animal a été attribué en début d'étude et identifie un animal dans une étude.

(**) Dans XYBION , les paramètres biologiques sont regroupés en déterminant , par exemple:
le déterminant DIFFERENTIAL CELL COUNT correspond aux paramètres:
- NEUTROPHILS (NEUT)
- EOSINOPHILS (EOSN)
- BASOPHILS (BASO)
- LYMPHOCYTES (LYMP)
- MONOCYTES (MONO)

En ce qui concerne le groupe , étant donné que nous ne conservons que les résultats de groupes témoins , le numéro de groupe sera toujours le même. En effet , il y a toujours un groupe témoin et les résultats de celui-ci sont d'office placés dans le même numéro de groupe. De plus , nous ne tiendrons pas compte du sous-groupe étant donné qu'il n'est pas utilisé.

2) du logiciel XYBANI : c'est l'équivalent de XYBION pour les données des animaux en quarantaine , c'est-à-dire ceux ne faisant pas encore partie d'une étude .
Par conséquent , il s'agira uniquement de valeurs de référence.

Les modalités de prélèvement des données sont les mêmes que pour XYBION. Seule l'interprétation des informations obtenues diffère :

- le code étude a été attribué sans avoir ici la signification qu'il a lors d'une étude réelle .
Il s'agit d'un code bidon dont le seul but est de pouvoir conserver les informations sur les animaux en quarantaine dans XYBANI;

- le numéro d'animal consiste en fait à associer le numéro de lot avec un numéro séquentiel;

- le jour de l'étude fera référence à la date d'entrée en quarantaine de l'animal;

- le groupe a toujours la même valeur puisqu'il ne peut s'agir que de témoins (il ne devra pas être considéré).

Contrairement à XYBION , il n'y a ici que les données relatives à des NON-RONGEURS (ceci est une conséquence de la numérotation individuelle de ceux-ci en quarantaine).

3) d'introductions manuelles: c'est l'utilisateur qui doit introduire toutes ses données pour l'étude.

2. Transfert vers la B.D.

A) Transfert de XYBION vers la B.D.

Le logiciel XYBION utilisant des formats de fichiers qui lui sont propres , il est nécessaire de convertir ces fichiers pour pouvoir manipuler aisément leur contenu.

Dans ce but, plusieurs possibilités s'offrent à nous. La première serait d'utiliser les extractions standards offertes par XYBION qui portent sur un seul paramètre à la fois . D'autre part , il existe un logiciel offrant un transfert par groupe de paramètres . Ce dernier n'est pas disponible à l'heure actuelle ce qui explique que la première solution servira de base à la réalisation de l'interface entre XYBION et la B.D.

C'est au moment de ce transfert que l'on doit offrir à l'utilisateur la possibilité de rejeter certains résultats ne pouvant être pris en compte ; ceci devant être laissé à l'appréciation du chercheur. Les raisons du rejet devront être conservées (dans un fichier archive) pour des justifications ultérieures.

Exemples de rejet :

- si l'on dispose de plusieurs mesures concernant un même animal effectuées à différents moments trop proches les uns des autres , il faut éviter de les prendre toutes en considération ce qui pourrait fausser la distribution de valeurs. En effet , si un prélèvement de sang était répété , les mesures risqueraient de faire apparaître des valeurs plus faibles pour certains constituants sanguins.

- il faut éviter de peupler plusieurs fois la B.D. avec des données déjà introduites (ceci est vrai pour quelqu'entrée que ce soit dans la B.D.).

De plus , afin de pouvoir rajouter des informations non présentes dans XYBION , il doit être offert la possibilité d'introduire d'autres données relatives à l'étude , aux animaux de celle-ci (date d'entrée en quarantaine, age, ...) , aux résultats correspondant à ces derniers (méthode d'analyse utilisée , date du résultat , ...).

B) Transfert de Xybani vers la B.D.

Les données recueillies au stade de la mise en quarantaine n'ont pas été transférées dans XYBION , mais néanmoins ont été stockées dans XYBANI. Présentes dans la B.D. , elles pourraient faire l'objet d'études.

Il faut prévoir également à ce niveau la possibilité de rejeter certaines données et de demander des informations complémentaires (cfr point précédent).

C) Entrée manuelle des résultats

On peut , par exemple , citer le cas où les données expérimentales ne seraient pas enregistrées du fait d'une défaillance lors des transferts des appareils de mesure vers XYBION , ce qui nécessiterait l'entrée manuelle des données.

A ce stade il faut prévoir pour toutes les données introduites un contrôle de validité. Celui-ci portera tant sur le format des entrées que sur la redondance éventuelle dûe à une introduction préalable déjà actée.

En outre , lors d'études , des échantillons pour la mise au point d'analyses sont parfois utilisés ; ceux-ci proviennent d'animaux en quarantaine bien identifiés.

Les valeurs obtenues avec ces échantillons sont susceptibles de participer à l'ensemble des données qui permettront d'établir une distribution. Ceci est particulièrement vrai dans le cas des singes pour lesquels peu de valeurs sont disponibles.

3. Choix des paramètres à stocker

Suite à un ensemble de discussions avec les utilisateurs potentiels , les paramètres suivants ont été sélectionnés :

- paramètres de sélection :
  * espèce ,
  * age ,
  * race (souche) ,
  * provenance (fournisseur) ,
  * sexe ,
  * voie d'injection ,
  * volume administré ,
  * méthode d'analyse ,
  * code de l'étude ,
  * durée de la période de pretest ,
  * période d'acclimatation des animaux ,
  * date d'entrée dans l'animalerie ,

* etat alimentaire (à jeun ou non) ,

* type d'aliment reçu ,

* type d'échantillon (serum,plasma ou urine) ,

- paramètres de biologie clinique : voir la liste officielle des paramètres de XYBION (annexe n°4)

- paramètres de croissance des animaux :

* poids corporel

* date de mesure du poids

- commentaires significatifs

4.3.2 Exploitation de la B.D.

1. Type d'information susceptible d'être obtenu à partir de la B.D.

La vocation de cette base de données est de pouvoir effectuer des traitements statistiques portant essentiellement sur les résultats disponibles.

** Pour ce qui est de la biologie clinique , un certains nombres de critères seront utilisés pour sélectionner exclusivement les résultats. Dans ce cas, on ne s'intéressera donc pas à des statistiques sur le nombre d'animaux , de telle espèce , etc... (bien que cette B.D. ne soit pas créée dans le but d'en faire ce type d'exploitation , il est tout à fait envisageable d'y apporter des extensions si nécessaire). Ces critères sont :

1. espèce

2. souche (race)

3. fournisseur

4. sexe

5. âge au moment de la mesure du résultat

6. date d'entrée dans l'animalerie

7. etat alimentaire (à jeun ou non)

8. type d'échantillon

9. nom du paramètre biologique

10. méthode d'analyse

REM : la méthode d'analyse doit être considérée car les valeurs normales peuvent changer suivant le type de méthode utilisé. Cela ne signifie pas qu'il faudra introduire absolument ce critère , par exemple , s'il on désire vérifier par une recherche l'hypothèse qu'il existe des différences significatives entre méthodes. Par ailleurs , ce sera l'utilisateur qui sera responsable de la classification des méthodes. Un changement de température , par exemple , pour une certaine méthode devra être considéré par l'utilisateur comme une méthode différente.

C'est chaque fois à partir de ces dix critères que se feront les demandes statistiques de biologie clinique. Il est entendu qu'il ne faudra pas nécessairement introduire des valeurs pour chacun de ces dix critères. Il faut cependant signaler que plus il y en aura de spécifiés , plus la recherche d'information sera ciblée et plus celle-ci sera rapide pour l'utilisateur.

** En ce qui concerne la toxicologie , il est davantage intéressant d' obtenir des statistiques portant sur l'évolution des résultats répertoriés au cours d'une étude. Par exemple , il est intéressant de connaître la variation de poids pour un animal donné et de la comparer avec l'évolution moyenne des individus.

Les critères sont :

1. code de l'étude

2. espèce

3. race (souche)

4. sexe

Suivant le type de demande de traitement statistique, l'un ou l'autre de ces critères doit être envisagé (voir point 2 ci-après).

2. Type de résultats à obtenir

Des résultats statistiques simples pourront être obtenus . Il est , par exemple , intéressant de pouvoir disposer dans tous les cas des renseignements tels que :

- la moyenne de l'échantillon
- la médiane
- l'écart-type , la variance

Une liste des demandes plus particulières a été établie pour la biologie clinique et pour la toxicologie :

** Biologie clinique :

1) taux d'un paramètre biologique en fonction de l'âge:

l'utilisateur désire obtenir une évolution du taux d'un paramètre en fonction de l'âge pour un certain nombre de critères (espèce,sexe,...). Le logiciel doit permettre d'offrir un ajustement de la courbe afin de pouvoir déterminer des valeurs de taux pour un âge donné. L'intervalle de confiance est souhaité dans la mesure du possible.

REM : en fonction du type de relation entre l'âge et le paramètre, il n'est pas toujours possible de formuler l'intervalle de confiance. S'il s'agit d'une régression linéaire , par exemple , cela ne pose pas de problème.

2) taux d'un paramètre en fonction du poids :

c'est le même type de demande qu'au point précédent pour le poids.

3) taux d'un paramètre en fonction du sexe :

il est demandé :

- de fournir la distribution des valeurs de ce paramètre séparément pour chacun des sexes. Une représentation graphique est souhaitée soit sous forme de courbe de Gauss , soit sous forme d'histogramme. Comme il le sera vu au point 4.4 , le choix de l'un ou l'autre type de représentation sera dicté par un test de normalité de l'échantillon obtenu par la recherche dans la B.D.

- de donner les valeurs de référence considérées comme normales . Une analyse sur celles-ci est demandée (celle-ci est présentée au point 4.3.3).

- de tester l'égalité des moyennes pour les mâles et pour les femelles.

Le même type de demande a été formulé pour :

4) taux d'un paramètre en fonction du type d'aliment

5) taux d'un paramètre en fonction du fournisseur

6) taux d'un paramètre en fonction de la race

7) taux d'un paramètre en fonction de l'espèce

8) taux d'un paramètre en fonction du type d'injection

** Toxicologie :

1) évolution du poids en cours d'étude :

l'utilisateur désire pouvoir comparer l'évolution d'un animal pour lequel il teste l'effet d'un produit avec l'évolution des animaux témoins d'une étude , le but étant de voir s'il y a ou non un effet provoqué par ce produit. Les données des animaux témoins proviennent de la B.D. et les données concernant l'animal testé seront introduites par l'utilisateur. Il faut également pouvoir fournir à celui-ci le poids attendu d'un animal témoin pour un jour de l'étude donné ; ce poids se calcule au moyen de l'équation théorique obtenue en ajustant les points obtenus par la recherche dans la B.D.

2) comparaison en fonction du sexe de l'évolution du poids en cours d'étude :

il est demandé en plus par rapport au point précédent de distinguer les mâles et les femelles.

3) évolution du poids en fonction de l'âge :

c'est le même type de demande qu'au point 1) mais l'utilisateur s'intéresse ici à voir l'évolution non pas pour une étude donnée mais pour toutes les études confondues. Il est clair que dans ce cas , il faudra introduire des critères de sélection pour cibler une espèce , une race ou un sexe déterminés. En effet , l'étude à laquelle il était référé pour les deux premiers types de demande en toxicologie portait sur une espèce et une race bien définies.

4) comparaison en fonction de la race de l'évolution du poids avec l'âge :

le principe est analogue au point 3) mais ici l'utilisateur veut voir l'évolution séparée des différentes races.

5) comparaison en fonction du sexe de l'évolution du poids avec l'âge :

de nouveau , il s'agit du point 3) où l'évolution pour chacun des deux sexes est envisagée séparément.

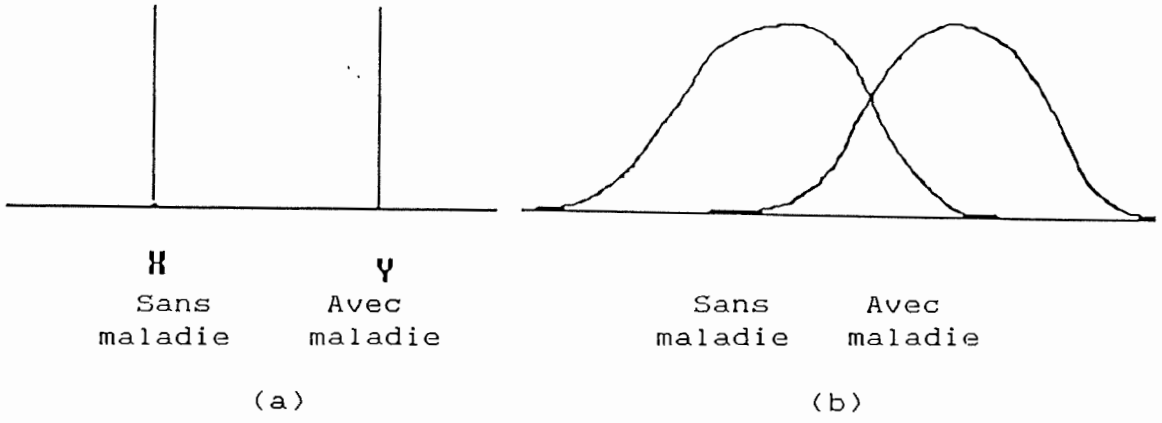
4.3.3 Analyse de la notion de "Valeurs de référence"

1. Le problème

Les composants de l'organisme humain et de tout organisme vivant en général sont sujets à des variations causées par les processus physiologiques , des différences génétiques , des maladies et par des facteurs environnementaux [SOLBERG 87a]. Les valeurs observées d'un constituant biologique chez un individu ne présentent d'intérêt que si elles sont comparées à des valeurs dites de référence , recueillies chez un même individu ou chez un groupe d'individus semblables définis par des critères d'exclusion ou de partition. Idéalement , une valeur observée devrait être comparée à des valeurs de référence obtenues chez le même individu [ALBERT 83] . En pratique toutefois , une telle approche n' est pas toujours possible et on se contente d'établir des valeurs de référence de groupe.

L'information contenue dans un ensemble de valeurs de référence est généralement synthétisée et communiquée à l'utilisateur sous la forme d'un intervalle de référence défini par deux limites , dites elles aussi de référence. Cette intervalle est destiné à mesurer l'étendue des valeurs pour un test pour la plupart des individus qui sont dépourvus de toute maladie. Définir un tel intervalle est un prérequis pour l'utilité d'un test [RIEGELMAN 80] .

La figure ci-après représente le problème tel qu'il le serait idéalement (a) et tel qu'il l'est en réalité (b) [RIEGELMAN 80] :



2. Définitions

Anciennement , la terminologie utilisée pour désigner les valeurs de référence faisait intervenir la notion de normalité. Il était question de "valeurs normales" , d'"intervalles normaux".

Le recours à cette notion pose cependant beaucoup de problèmes et on lui préfère la notion de référence. Dans la littérature , on retrouve les deux terminologies qui en réalité touchent au même concept utilisé pour l'interprétation des tests de biologie clinique [BOYD 83].

Une définition des "valeurs de référence" est proposée par Dybkaer et semble être communément acceptée [SUNDERMAN 75],[GALEN 80]:

> " Un ensemble de valeurs d'un certain type de quantité obtenu à partir d'un seul individu ou d'un groupe d'individus correspondant à une description fixée. Cette description doit être déchiffrée et disponible si d'autres veulent utiliser les valeurs de référence Pour chaque type de quantité , une série de groupes de références sera nécessaire , prenant en considération l'âge , le sexe , la race , etc..."

Deux notions méritent d'être définies [CP 89],[BARR 81]

* Sensibilité d'un test : c'est la proportion de vrais positifs par rapport à l'ensemble de la population malade.

$$\text{SENSIBILITE} = \frac{\text{VRAIS POSITIFS}}{\text{POPULATION MALADE}}$$

Elle se réfère à la capacité qu'a un test de détecter des patients avec la maladie en question et diminue si on prend un seuil de rejet plus faible , par exemple = 1 % au lieu de 5 % .

* Spécificité d'un test : c'est la proportion des vrais négatifs par rapport à l'ensemble de la population saine.

$$\text{SPECIFICITE} = \frac{\text{VRAIS NEGATIFS}}{\text{POPULATION SAINE}}$$

Elle correspond à la probabilité qu'un test a de faire apparaître des résultats non significatifs pour des individus sains. La spécificité augmente si on prend un seuil de rejet plus faible (cfr ci-dessus).

3. Facteurs influençant la distribution d'un échantillon

Pour établir des groupes de référence , il est important de prendre en compte les facteurs qui peuvent influencer la distribution , à savoir :

1. Taille de l'échantillon
2. L'age
3. Le sexe
4. Le poids
5. L'état de santé
6. Facteurs raciaux et génétiques
7. Les variations temporelles (jour/nuit et les saisons)
8. Les habitudes diététiques
9. Le stress
10. L'activité physique
11. Le cycle menstruel (femelles)
12. La grossesse (femelles)
13. Les médicaments
14. La collection de l'échantillon et les traitements subis par celui-ci.

Cette liste [ASH 80] est adaptée plus particulièrement aux tests effectués sur des êtres humains mais peut être extrapolée aisément pour les animaux (les résultats présents dans notre B.D. s'y rapportant).

La détermination de "valeurs de référence" peut être biaisée par certains autres facteurs [CP 89] :

- les individus pris en considération peuvent cacher des maladies non détectées et fausser la distribution.

- les "valeurs de référence" sont parfois calculées à partir d'un nombre trop faible pour être statistiquement fiable.

- les valeurs obtenues par une méthode peuvent être inappropriées pour une autre méthode.

- les valeurs fournies par les fabriquants de kits de test ne correspondent pas assez fréquemment aux résultats obtenus sur une population locale par un laboratoire local , parfois sans aucune raison démontrable. Le même problème est rencontré avec les "valeurs normales" obtenues dans la littérature.

- les valeurs d'une population peuvent être non distribuées aléatoirement et être biaisées à une ou l'autre extrémité de l'intervalle. Ceci affecte le calcul de l'écart type et déforme l'"intervalle de référence". Il est dans ce cas préférable d'utiliser des méthodes non paramétriques (voir point 5,6 ci-après).

4. Principes fondementaux

Il est cinq principes qu'il faut absolument comprendre pour pouvoir bien interpréter les résultats de tests [RIEGELMAN 80].

a. principe1 : "En dehors des limites normales" n'égale pas "malade".

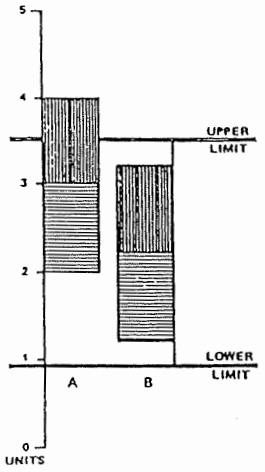
Du fait que 5 % d'individus non malades sont souvent définis comme étant en dehors des limites normales , il n'est pas exact (voir méthodes paramétriques point 3.3 $2) d'égaler "en dehors des limites normales" avec "anormal". De plus , cela peut être du à une imprécision de la technique utilisée. Par exemple , si une personne se trouve à la limite supérieure de ce que l'on nomme "normal" , une imprécision de la méthode peut faire passer la limite établie.

b. principe2 : "A l'intérieur des limites normales" n'égale pas "sans maladie".

Beaucoup de tests n'ont pas une bonne aptitude à séparer les individus malades et non malades. De plus , certains tests sont révélateurs pour certaines maladies et pas pour d'autres.

c. principe3 : Des changements au sein des limites normales peuvent être pathologiques.

Voici un exemple pour illustrer ce principe. Supposons deux patients A et B avec des valeurs normales (quand ils ne sont pas atteints de la maladie testée) fort distinctes. La valeur du patient A se trouve proche de la limite supérieure alors que la valeur du patient B est située au milieu de l'intervalle de référence. Une augmentation similaire de chacune des deux valeurs ne met en évidence une valeur hors limite que pour le patient A alors que le patient B est également atteint de la maladie. La figure ci-après représente l'exemple [CP 89].



L'effet sur le patient B n'a donc pas été mis en évidence. Ceci est du au fait que l'intervalle de référence est assez large. Des données sur l'individu B auraient été préférables à cet intervalle de référence obtenu à partir d'une population !

d. principe4 : L'intervalle de référence est lié à une population de référence.

Ceci est en rapport avec les facteurs influençant la distribution d'un échantillon qui ont été mentionnés précédemment.

e. principe5 : "Dans les limites" n'égale pas "désirable".

Certaines valeurs peuvent être plus hautes que souhaitables et peuvent indiquer une tendance croissante des membres d'une population de référence au développement d'une maladie dans le futur. Par exemple , le poids et le taux de cholestérol des Américains sont souvent dits être , en moyenne, plus hauts que souhaitable et contribueraient au développement des maladies du coeur.

5. Choix d'un type de méthode pour la détermination des valeurs de référence

Plusieurs types d'intervalles de référence ont été proposés : intervalle inter-quantiles , intervalle de prédiction , intervalle de tolérance [ALBERT 83]. Quoi que d'un point de vue purement statistique , ces intervalles possèdent des propriétés bien définies , les différences numériques entre ces intervalles s'amenuisent , dès lors que l'on dispose d'un nombre suffisant de valeurs de référence. Il faut garder à l'esprit que l'intervalle de référence sert surtout de guide pour fournir une interprétation des valeurs observées.

Les intervalles de référence inter-quantiles sont les plus fréquemment utilisés , car ils sont faciles à déterminer que ce soit par des méthodes paramétriques ou non paramétriques.

Conventionnellement , on admet que l'intervalle de référence inter-quantiles contient une proportion de 95 % de la population de référence. Ceci provient d'une convention de prendre 5 % comme niveau d'incertitude pour les épreuves statistiques. Cependant , si l'on désire obtenir 99 % de la population de référence , on peut utiliser les quantiles 0.995 et 0.005 qui pour une distribution gaussienne donne 2.58 et -2.58 respectivement.

Cette façon de procéder définit des intervalles de référence symétriques.

L'estimation des quantiles peut se faire par des méthodes paramétriques ou non paramétriques. Les méthodes paramétriques supposent que les valeurs de référence proviennent d'une distribution de type connu (normale , le plus souvent). Les techniques non paramétriques ne formulent aucune hypothèse sur la distribution des valeurs de référence et sont donc toujours applicables. L'estimation paramétrique des quantiles est en principe plus précise , surtout pour de petits échantillons , pour autant bien entendu que les valeurs de référence proviennent de la distribution dont on suppose le type connu.

Avant de parler du choix d'un type de méthode à adopter , il convient de donner un mot concernant les valeurs aberrantes. Celles-ci doivent parfois être éliminées , ou les expériences à nouveau menées. Dans la conception de notre B.D. , l'utilisateur qui introduira les données aura la possibilité de rejeter certaines d'entre elles ce qui justifie qu'on ne tentera pas d'éliminer des valeurs.

Le type de méthode à appliquer dépend de la distribution à laquelle on a à faire. Lorsque celle-ci n'est pas normale, on recourt plutôt à une méthode non paramétrique qui ne fait aucune hypothèse sur la distribution. Dans le cas contraire , une méthode paramétrique sera plutôt envisagée. Comme il a été explicité au point 3.3.d , un test de normalité est réalisé pour prendre une décision sur le choix de la méthode à suivre.

Il faut cependant signaler que des transformations peuvent être réalisées afin de rendre une ditribution normale. Par exemple , la transformation "logarithmique" est fréquemment utilisée et est particulièrement adaptée à une distribution asymétrique à droite (fig 24a). Il en est de même pour la transformation "racine carrée". Pour une asymétrie à gauche , on utilisera la transformation "puissance" ou "au carré" (fig 24b) [ALBERT 83].

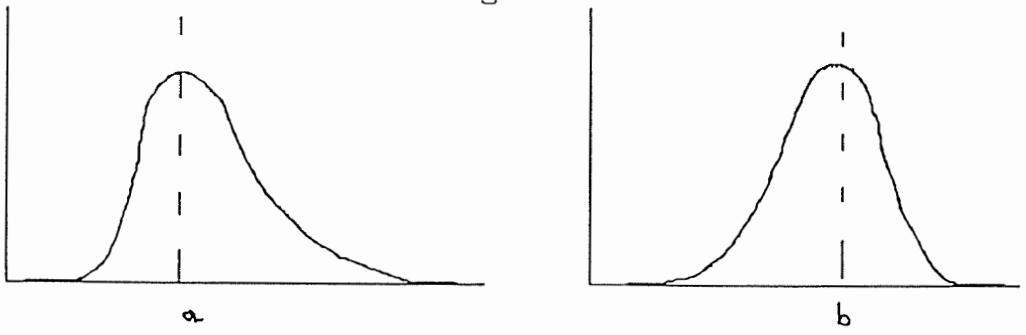
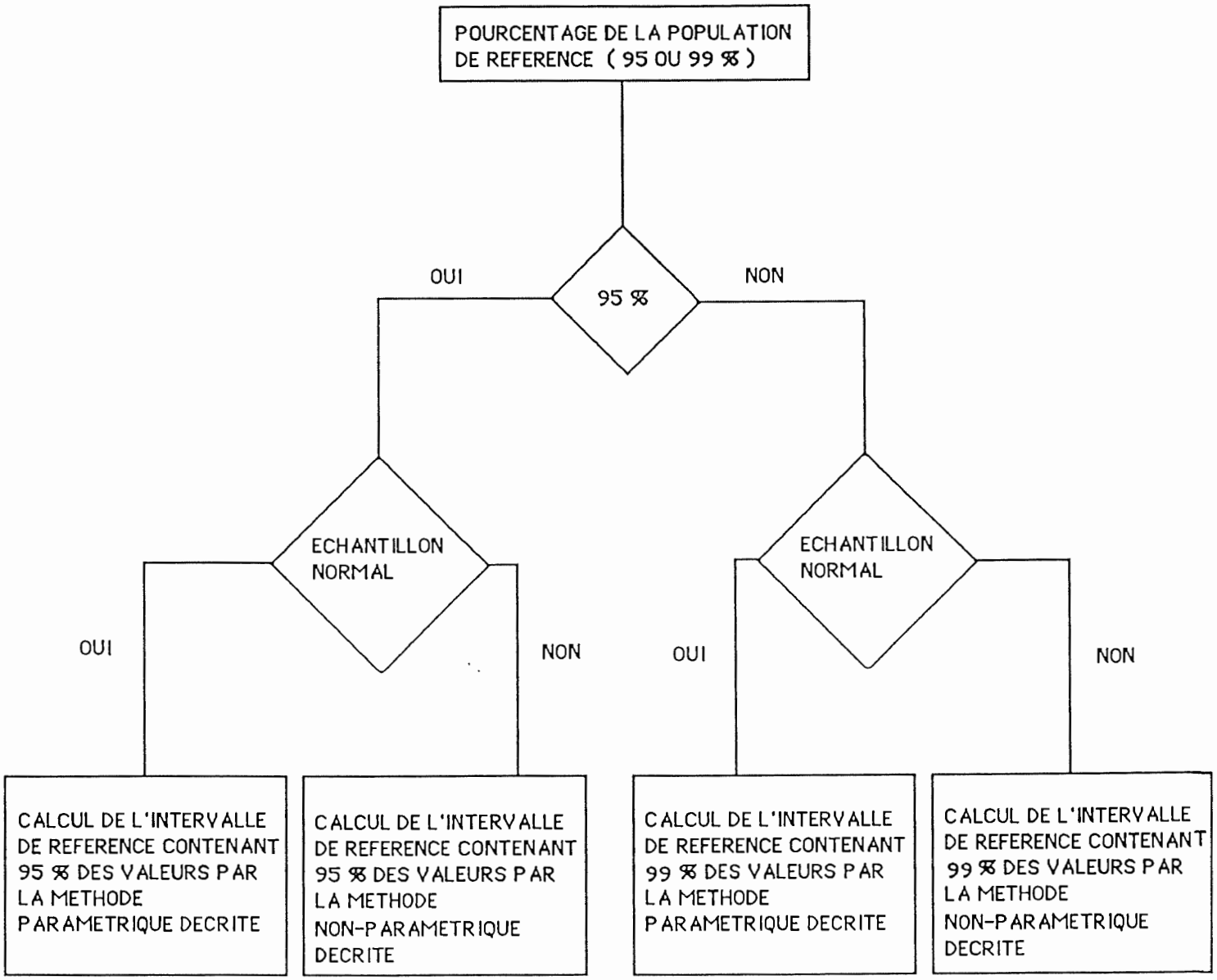


Figure 24 : Distributions asymétriques [ALBERT 83]

D'autres types de transformation ont été rapportés [SHULTZ 85] et certaines d'entre elles semblent s'adapter mieux que les transformations classiques (notamment logarithmique).

6. Implémentation de ces concepts

Dans notre logiciel , un sous-module du module "Traitements statistiques" (voir point 4.4.3.B.) est responsable de la détermination des valeurs de référence. L'organigramme présenté ci-après fait la synthèse de la démarche envisagée :



Les méthodes utilisées sont celles détaillées au point 3.3§2.G.

7. Liste des requêtes demandant ce calcul des valeurs de référence

Ce calcul est réalisé pour :

1) taux d'un paramètre en fonction du sexe

2) taux d'un paramètre en fonction du type d'aliment

3) taux d'un paramètre en fonction du fournisseur

4) taux d'un paramètre en fonction de la race

5) taux d'un paramètre en fonction de l'espèce

6) taux d'un paramètre en fonction du type d'injection

Pour plus de détail sur ces requêtes , on peut se référer au point 4.3.2 "Exploitation de la B.D." §2 "Biologie clinique".

4.4 LA CONCEPTION LOGIQUE.

4.4.1. INTRODUCTION.

Contrairement à ce qui se fait habituellement avec un langage de troisième génération, le cycle de développement d'une application réalisée avec un langage de quatrième génération ne comprends pas nécessairement les différentes étapes classiques de réalisation d'un projet (cfr.chap.3.1).

De plus, vu l'environnement de développement dans lequel nous nous trouvions, nous avons utilisé, pour la majeure partie du travail, la méthode adoptée sur place. C'est ainsi que suite au "rapport de faisabilité" et au "cahier des charges" qui l'a précédé, nous avons pu commencer la conception proprement dite.

Pour celle-ci, nous avons utilisé dans une premiére étape les méthodologies de développement qui nous ont été enseignées à l'Institut . Le choix de celles-ci est dû à leur relative facilité d'utilisation et à l'avantage de clarté qu'elles peuvent offrir pour aborder, de manière générale, un problème posé.

Dans un premier temps, nous avons réalisé un schéma entité/association [BODART 88], qui est un modèle d'analyse conceptuelle, indiquant clairement et sous forme visuelle, le contenu sémantique du problème ainsi que les inter-relations existantes entre les différents constituants ou concepts.

A partir de celui-ci, nous avons utilisé la méthodologie du MAG [HAINAUT 86], qui est un modèle de spécifications de structures de données, afin de dégager certaines structures et de comprendre les mécanismes à mettre au point pour traîter les informations disponibles.

Le schéma qui en a découlé nous a alors permis d'établir l'approche définitive du contenu logique du travail. Il a été suivi par la conception physique elle-même et par l'implémentation en Powerhouse.

Il faut souligner ici que vu la nature du logiciel utilisé ainsi que l'environnement de développement (contraintes propres à l'entreprise) dont nous avons discuté ci-dessus, nous n'avons pas pu appliquer en toute rigueur les principes standards de développement d'un projet informatique.

4.4.2. ETUDE DE LA SEMANTIQUE DU PROBLEME.

A. Schéma Entité/Association.

L'étude de la sémantique du problème (l'analyse conceptuelle), nous a amené à construire le schéma Entité/Association représenté à la figure 26.
Il représente une structuration possible des informations disponibles.

Pour rappel et sans entrer dans les détails, celui-ci se base sur les principes suivants.

Tout objet ou concept individualisé du réel est dénommé "Entité". Les entités sont répertoriées en classes appelées "Type d'entité". Elles peuvent être en association les unes avec les autres ; ces association étant classées en "Type d'association". Chacun de ces types d'association précise, en outre, le nombre, le type et le rôle des entités impliquées dans une association de ce type.

Quant aux propriétés des objets du réel, elles sont représentées par des "attributs". Ceux-ci sont reliés aux types d'entité et aux types d'association dont ils représentent les propriétés.

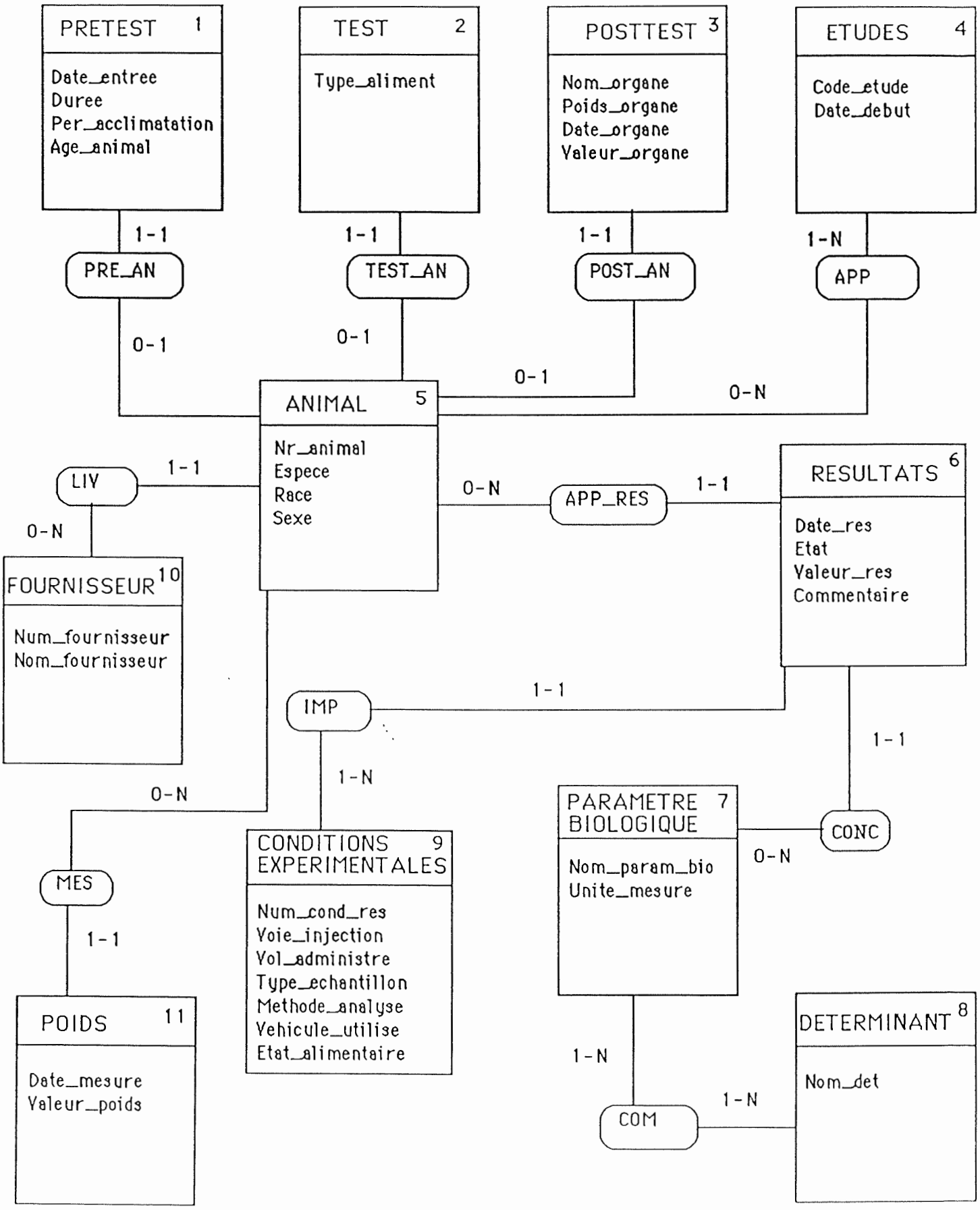
Graphiquement, ces différentes composantes se représentent de façon distincte. Une entité est représentée par une boîte étiquetée au nom de l'entité. Une association est représentée par une boîte arrondie reliée à deux entités. Les attributs sont disposés sous les types d'entité ou d'association auxquels ils se rapportent.

Enfin, par un couple de chiffres apposés sur les lignes reliant des entités, on indique le nombre de fois qu'une entité donnée (la plus proche du couple de chiffres) participe à l'association qui lui est reliée. On parle de "connectivité". Ceci se représente par le nombre minimum de participation à une association que peut avoir l'entité considérée, suivi du nombre maximum de ces participations.

Par exemple, on trouvera les couples de chiffres suivants :

- 1 - 1 : indique qu'une entité participe toujours à une et une seule association de ce type.
- 0 - N : indique qu'une entité peut participer à une ou plusieurs associations de ce type.
- 1 - N : indique qu'une entité participe toujours à une ou plusieurs associations de ce type.

Figure 26 : Schema entite/association



Selon cette représentation graphique, voici le schéma entité/association du problème (cfr.fig.26).

Commentaire et justification.

Suite à l'énoncé du problème (cfr.chap.4.3), nous avons pu distinguer les types d'entités suivants :

- T.E. PRETEST :

Il répertorie toutes les informations dont on veut disposer en ce qui concerne les phases Prétest et quarantaine d'une étude de toxicologie.

Les données se rapportant à ces deux phases ont été regroupées dans le même type d'entité du fait que certaines données de quarantaine (durée, période d'acclimatation) ne sont connues qu'en phase pretest. Il semblait dés lors plus intéressant de les regrouper avec ces données de Prétest.

ATTRIBUTS:
- date_entrée: date d'entrée d'un animal en quarantaine.
- per_acclimatation: durée de la période d'acclimatation d'un animal avant étude.
- durée: durée de la quarantaine pour un animal donné
- age_animal: âge de l'animal au jour de la date d'entrée dans l'animalerie.

IDENTIFIANT:

une entité du type Prétest est identifiée par l'ensemble de ses attributs et par le rôle joué par le type entité Animal dans l'association (PRE_AN) avec cette entité du type Prétest.

CONNECTIVITE:

ce type d'entité participe à l'association PRE_AN avec une connectivité (1-1).

- T.E. TEST :

Il répertorie toutes les informations dont on veut disposer en ce qui concerne la phase test d'une étude de toxicologie.

ATTRIBUTS:
- type_aliment: c'est le type d'aliment donné à un animal en cours d'étude.

IDENTIFIANT:
une entité du type Test est identifiée par l'ensemble de ses attributs et par le rôle joué par le type entité Animal dans l'association (TEST_AN) avec cette entité du type Test.

CONNECTIVITE:
ce type d'entité participe à l'association TEST_AN avec une connectivité (0-1).

- **T.E. POSTTEST :**
Il répertorie toutes les informations dont on veut disposer en ce qui concerne la phase posttest d'une étude de toxicologie.

ATTRIBUTS:
- poids_organe: c'est le poids d'un organe pour un animal.
- nom_organe: c'est le nom d'organe pour lequel on a effectué une mesure de poids.

IDENTIFIANT:
une entité du type Posttest est identifiée par l'ensemble de ses attributs et par le rôle joué par le type entité Animal dans l'association (POST_AN) avec cette entité du type Posttest.

CONNECTIVITE:
ce type d'entité participe à l'association POST_AN avec une connectivité (0-1).

- T.E. ANIMAL :
répertorie les attributs propres à un animal donné qui séjourne dans l'animalerie.

ATTRIBUTS:
- nr_animal: c'est le numéro attribué à l'animal et permettant de l'identifier.
- espece: nom de l'espèce à laquelle appartient l'animal.
- race: nom de la race à laquelle appartient l'animal.
- sexe: sexe de l'animal.

IDENTIFIANT:
nr_animal

CONNECTIVITES:
- dans l'association PRE_AN : connectivité (0-1)
- dans l'association TEST_AN : connectivité (0-1)

- dans l'association POST_AN : connectivité (O-1)
- dans l'association LIV : connectivité (1-1)
- dans l'association APP : connectivité (O-N)
- dans l'association MES : connectivité (O-N)
- dans l'association APP_RES : connectivité (O-N)

Le quatrième type de rôle joué par une entité du type Animal est obligatoire; c'est-à-dire qu'une entité animal ne peut exister sans participer à une association du type "Liv".

Les six autres reflètent le caractère facultatif de la participation d'une entité Animal à une association du type "Pre_an","Mes","App_res","App","Test_an" et "Post_an".

- T.E. RESULTATS:

Reprend les attributs concernant les résultats observés sur un animal donné au cours de sa vie (depuis l'entrée dans l'animalerie jusqu'à sa mort).

ATTRIBUTS:
- date_res: date de mesure du résultat observé pour un animal donné.
- etat: phase dans laquelle se trouve l'animal au moment de la mesure du résultat (phase Test,Pretest,Posttest ou Quarantaine).
- valeur_res: valeur du résultat observé pour l'animal.
- commentaire: commentaire facultatif que l'expérimentateur pourrait faire au sujet du résultat observé.

IDENTIFIANT:

une entité de ce type est identifiée par l'ensemble de ses attributs et par le rôle joué par le type d'entité Animal dans l'association (APP_RES) liée à une entité du type Résultats ainsi que par le rôle joué par le type d'entité Paramètre-biologique dans l'association (CONC).

CONNECTIVITES:
- dans l'association APP_RES: connectivité (1-1)
- dans l'association IMP: connectivité (1-1)
- dans l'association CONC: connectivité (1-1)

T.E. PARAMETRE_BIOLOGIQUE:

c'est un répertoire des différents paramètres biologiques existants.

ATTRIBUTS:
- nom_param_bio: nom du paramètre biologique.
- unité_mesure: unité de mesure propre au paramètre biologique.

IDENTIFIANT:
nom_param_bio
CONNECTIVITE:
- dans l'association CONC: connectivité (O-N)
- dans l'association COM: connectivité (1-N)

Un paramètre biologique peut exister sans qu'il y ait aucun résultat le concernant, mais doit obligatoirement appartenir à un déterminant.

T.E. DETERMINANT:

C'est un répertoire des déterminations existantes dans la terminologie du logiciel XYBION. Chaque "déterminant" comprend un ou plusieurs paramètres biologiques.

ATTRIBUTS:
- nom_det: c'est le nom du déterminant.

IDENTIFIANT:
nom_det

CONNECTIVITE:
ce type d'entité participe à l'association COM:
connectivité (1-N).

Un déterminant n'existe pas sans qu'il soit relié à au moins un paramètre biologique.

T.E. CONDITIONS_EXPERIMENTALES:

C'est l'ensemble des informations dont on dispose à propos des conditions expérimentales que l'on peut observer. Par convention, il a été décidé qu'un numéro de condition expérimentale serait attribué à chacune d'entre elles.

ATTRIBUTS:
- voie_injection: type de voie d'injection utilisé pour l'analyse expérimentale.
- vol_administre: volume de substance administré à un animal.
- type_echantillon: type d'échantillon utilisé pour l'analyse.
- méthode_analyse: méthode analyse utilisée par l'expérimentateur. Par convention avec les utilisateurs, il a été décidé que chaque type de méthode d'analyse serait identifié par les utilisateurs eux-mêmes.
Ils attribuent eux-mêmes les noms des méthodes d'analyse pouvant exister.

- vehicule_utilise: véhicule utilisé lors du test expérimental.
- etat_alimentaire: état alimentaire de l'animal (AJEUN ou NOURRI).
- num_cond_res: numéro de condition expérimentale permettant d'identifier les différentes conditions expérimentales.

IDENTIFIANT:
num_cond_res

CONNECTIVITE:
dans l'association IMP : connectivité (1-N).
Une condition expérimentale ne peut exister en elle-même sans référer au moins un résultat.

T.E. POIDS:
concerne les informations relatives au poids de l'animal.

ATTRIBUTS:
- date_mesure: date de mesure du poids.
- valeur_poids: c'est la mesure du poids d'un animal à une date donnée.

IDENTIFIANT:
une entité de ce type est identifiée par l'ensemble de ses attributs et par le rôle joué par le type d'entité Animal dans l'association MES avec ce type d'entité Poids.

CONNECTIVITE:
dans l'association MES : connectivité (1-1).
Une entité Poids ne peut exister indépendamment d'un animal pour lequel on a mesuré le poids.

T.E. FOURNISSEUR:

répertoire des fournisseurs d'animaux de l'animalerie.

ATTRIBUTS:
- nom_fournisseur: nom du fournisseur.
- num_fournisseur: numéro de fournisseur.

IDENTIFIANT:
num_fournisseur

CONNECTIVITE:
dans l'association LIV : connectivité (0-N).
Un fournisseur peut être répertorié sans qu'il ait fourni d'animal.

T.E. ETUDES:
répertoire de toutes les études existantes.

ATTRIBUTS:
- code_etude: code de l'étude attribué par le logiciel XYBION.
- date_debut: date du début de l'étude.

IDENTIFIANT:
code_etude

CONNECTIVITE:
dans l'association APP : connectivité (1-N).

CONTRAINTES FONCTIONNELLES SUPPLEMENTAIRES :

- Pour les données de la phase QUARANTAINE :
  + la date entrée de quarantaine (d'une entité du type d'entité PRETEST) doit être égale à la date du début de l'étude. En effet, une étude correspondant à ces données de quarantaine a reçu un code d'étude bidon et a commencé au moment où les animaux sont entrés en quarantaine!

  + la date de chaque résultat (du type d'entité RESULTATS) doit être supérieure ou égale à la date du début d'étude.

- Pour les données de la phase PRETEST :
  + la date entrée de quarantaine (d'une entité du type d'entité PRETEST) doit être antérieure à la date du début de l'étude.

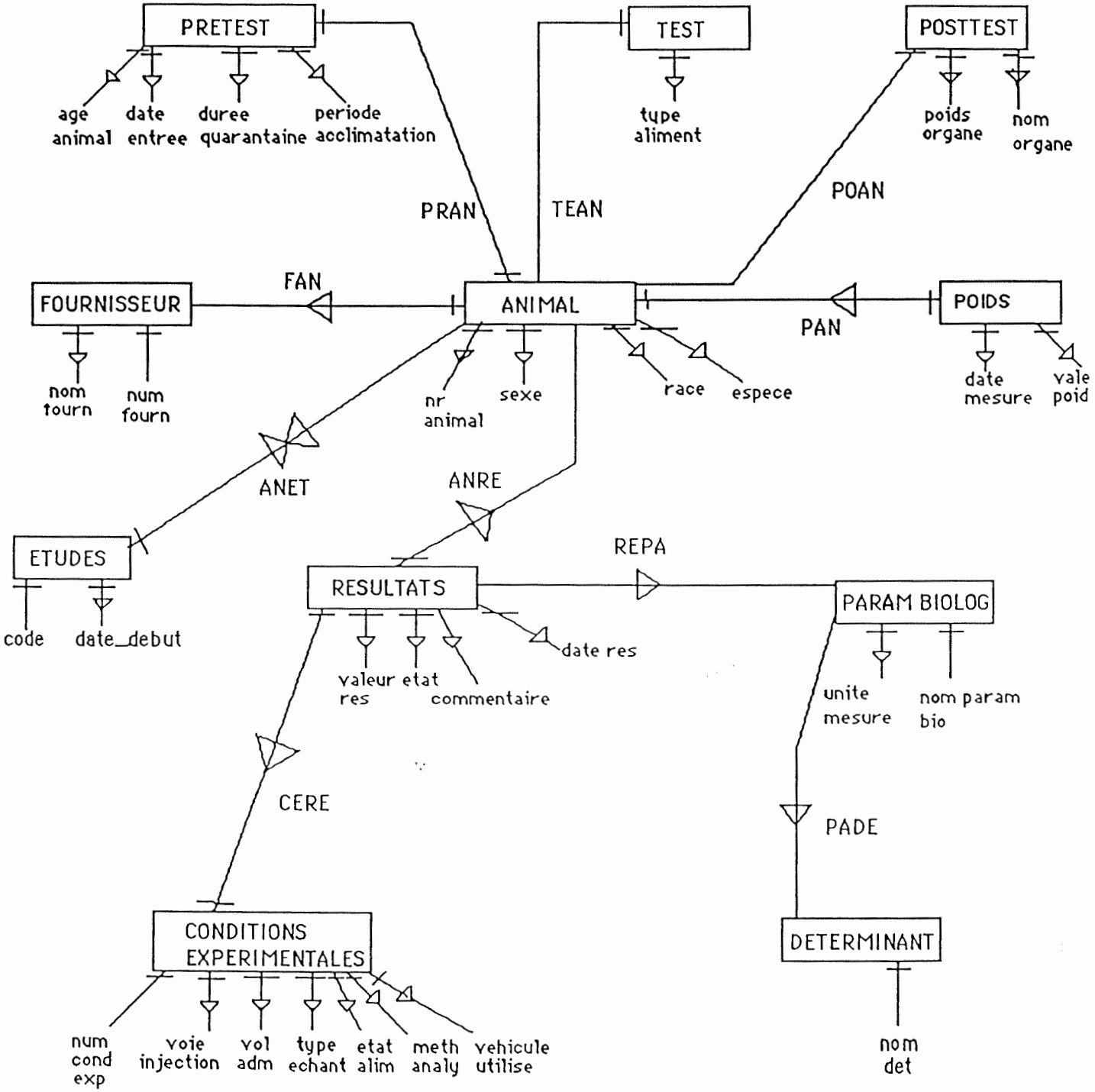
  + la date de chaque résultat (du type d'entité RESULTATS) doit être antérieure à la date du début de l'étude et postérieure à la date d'entrée en quarantaine.

- Pour les données de la phase TEST :
  + la date d'entrée en quarantaine (d'une entité du type d'entité PRETEST) doit être antérieure à la date du début de l'étude.

  + la date de chaque résultat (du type d'entité RESULTATS) doit être égale ou postérieure à la date d'étude.

- une entité du type ANIMAL doit, pour exister, participer à au moins une des associations PRE_AN ou TEST_AN ou POST_AN.

Figure 27 : Schema des accès possibles



remarque :

Dans la suite du travail, on ne considérera pas le type d'entité Posttest. En effet, après discussion avec les utilisateurs, il n'est pas apparu nécessaire de traiter ces informations pour le moment.

B. Schéma MAG.

A partir du schéma entité/association, nous pouvons par transformation, en déduire un schéma MAG qui a l'avantage de permettre une structuration des données claire et rigoureuse.

Le modèle MAG décrit les structures de données non seulement sous l'angle de la sémantique qu'expriment ces données mais aussi sous celui des accès dont elles peuvent faire l'objet. Il permet aussi de transformer les structures d'accès aux données de manière systématique.

Les concepts du modèle sont typiquement ceux que le programmeur d'application a coutume d'utiliser; certains d'entre eux, se prêtant à une représentation graphique, sont utilisés ici.

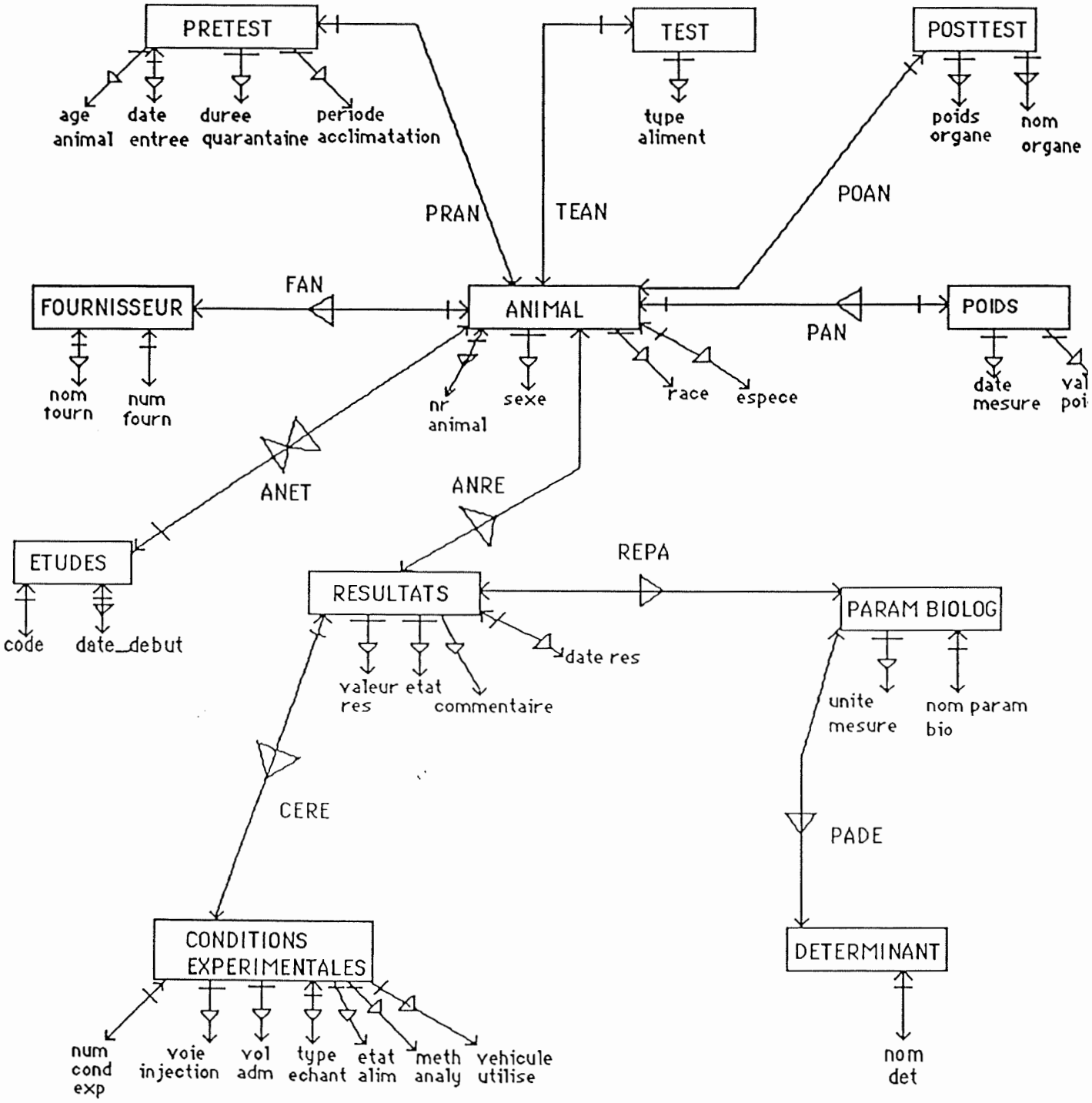
Les objets de base du modèle MAG que nous utilisons sont les articles et les types d'articles, les items et les valeurs d'item, les chemins d'accès, les fichiers et les clés d'accès [HAINAUT 86].

Le schéma MAG représenté à la figure 27 , est dérivé du schéma entité/association exposé précédemment. Il décrit l'ensemble des accès possibles entre entités.

On y observe les types d'article avec les items s'y rattachant et les chemins d'accès. Graphiquement, l'item est représenté par son nom, le caractère obligatoire d'un item est représenté par un trait horizontal apposé sur la barre verticale reliant l'item; on distingue aussi un item simple identifiant (pas de symbole), un item simple non identifiant (triangle inversé), un item répétitif identifiant (triangle) et un item répétitif non identifiant (diabolo).

Pour plus de précision quant à la méthode utilisée, on peut se rapporter aux ouvrages traitant du sujet [HAINAUT 86].

Figure 28 : Schéma des accès nécessaires



Commentaire et justification :

Le schéma MAG possède exactement la même sémantique que le schéma entité/association dont il dérive. Les types d'association ont disparu au profit de chemins liant deux entités. Chaque chemin est libellé avec un nom permettant de l'identifier.

Chaque attribut se voit affecté le caractère obligatoire. Ceci découle de l'énoncé du problème.

C. Le schéma des accès nécessaires.

Suite au schéma des accès possibles, il est nécessaire de mettre en évidence le type d'accès et donc les clés d'accès qu'il faut prévoir pour l'utilisation de la base de données.

Ceci se réalise en étudiant au cas par cas chaque type de requête qu'il est possible d'envisager à partir de l'énoncé du problème. On ne considère ici que les types de requête qui seront demandés par les utilisateurs.

On identifie ainsi les attributs (items) qui ne servent éventuellement pas dans tous ces types de requête (et qu'on peut alors supprimer), ainsi que les clés d'accès qui sont réellement nécessaires.

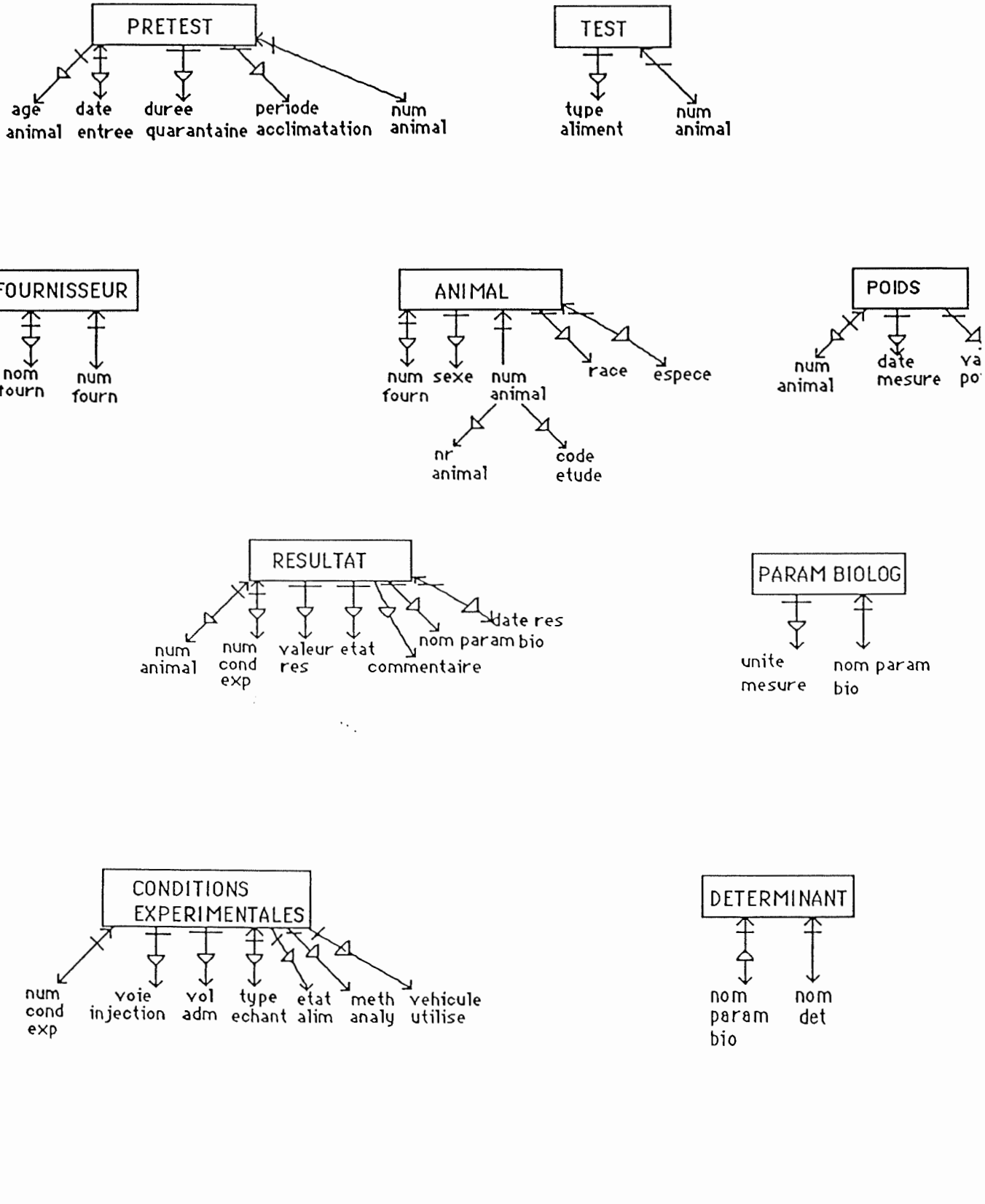
Le schéma obtenu est le schéma des accès nécessaires. Il est représenté à la figure 28.

D. Schéma compatible avec le système Powerhouse.

Le schéma des accès nécessaires nous permet de dériver un schéma conforme aux règles et principes de Powerhouse. Cette dérivation se fait par une série de transformations effectuées sur le schéma initial [HAINAUT 86].

Dans notre cas, il faut principalement éliminer les types de chemins. On peut à cette occasion rencontrer le problème du type d'articles ne possédant pas d'identifiant et devant être référencé par un autre type d'articles. De telles situations sont en effet créées par les transformations qui permettent d'éliminer des types de chemin [HAINAUT 86] .

Figure 29 : SCHEMA COMPATIBLE POWERHOUSE



On est alors amené à attacher à ce type d'articles un identifiant artificiel. Pour ce qui est de l'identifiant du type d'article Animal, nous avons adopté le mécanisme suivant.

Le numéro d'animal n'identifiant pas à lui seul un animal parmi l'ensemble des animaux existant ou ayant existés au cours de toutes les études réalisées jusqu'à présent, nous lui avons joint le code de l'étude auquel l'animal correspond.

L'item ainsi constitué du numéro d'animal + le code étude est , lui, identifiant pour un animal. Il est nommé "num_animal". De plus, comme dit au chapitre précédent, il faut savoir qu'un animal ne faisant pas encore partie d'une étude s'est vu attribuer un code étude "bidon" permettant de l'identifier (cas des non rongeurs seulement).

En outre, comme il sera rappelé plusieurs fois vu la complexité de la numérotation disponible, le mécanisme d'identification d'un animal est le suivant :

Pour les rongeurs :
- un numéro individuel + un code d'étude définitif identifie un animal en phase PRETEST.
- un numéro définitif + un code d'étude définitif identifie un animal en phase TEST.

Pour les non rongeurs :
- un numéro individuel + un code étude "bidon" identifie un animal en QUARANTAINE.
- un numéro individuel + un code étude définitif identifie un animal en phase PRETEST.
- un numéro définitif + un code d'étude définitif identifie un animal en phase TEST.

Le schéma transformé et rendu compatible avec le système Powerhouse est représenté à la figure 29.

4.4.3. CONCEPTION MODULAIRE PROPOSEE.

En se basant sur l'approche langage de quatrième génération, nous avons conçu le système en fonction des possibilités de prototypage offertes.

Pour rappel, un prototype permet de montrer rapidement aux utilisateurs l'aspect futur de l'application et d'en ajuster les fonctionnalités selon les besoins.

Nous avons pour ce faire découpé le projet en une série de modules. Chaque module se dégage par le fait qu'il concerne un même ensemble de concepts identiques ou bien par le fait que ces constituants assurent une même fonctionnalité.

Cette approche modulaire a l'avantage de pouvoir rapidement montrer un ensemble cohérent de la réalisation partielle du projet et faisant intervenir une partie seulement des modules existants.
De plus, elle offre une guarantie de maintenance beaucoup plus facile (si une fonctionnalité s'avère ne pas être remplie, il suffit de corriger uniquement le module qui doit l'assurer) et permet aussi de compléter plus tard l'architecture sans bouleverser tout ce qui existe.

Comme dit précédemment, le schéma général du problème nous a suggéré deux parties dans le projet :

- la constitution de la base de données, comprenant le transfert des données, la conception de la base de données et tous les problèmes liés aux conversions (interfaces).

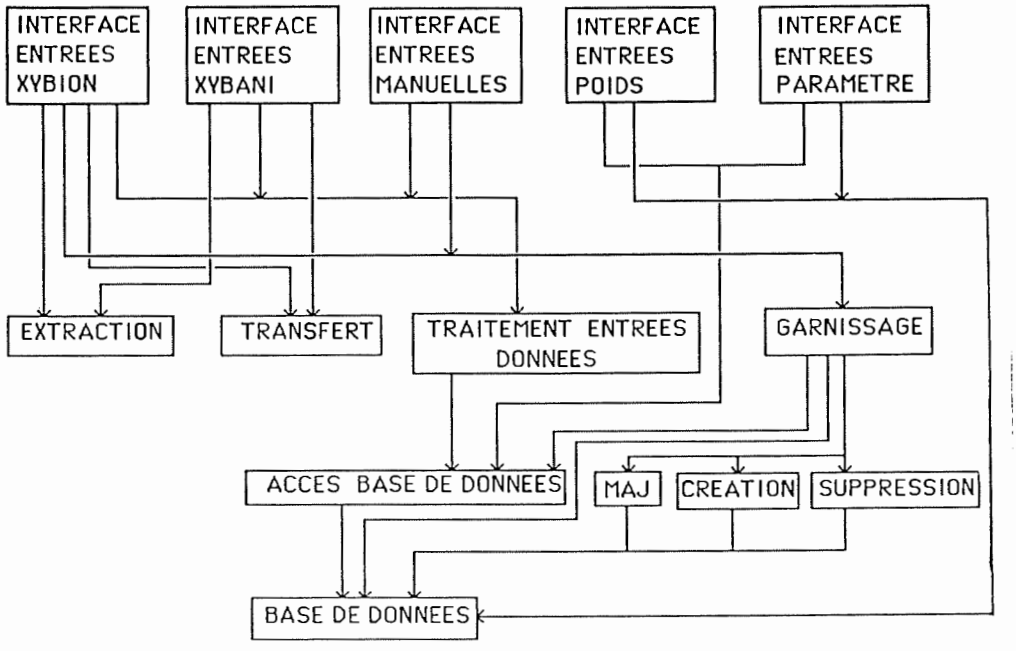
- l'exploitation de la base de données.

Ces deux parties feront l'objet d'une étude séparée bien que faisant partie de la même architecture générale. Ceci, uniquement pour des raisons de clarté.

A. PREMIERE PARTIE : CONSTITUTION DE LA B.D.

1° Schéma .

L'architecture logique proposée est représentée ci-dessous.



2° Description des différents constituants de l'architecture.

L'architecture proposée au point 1° découle de l'analyse que nous avons réalisée à partir de l'énoncé du problème.

La mise en évidence des différents modules proposés est justifiée ci-après. Mais il est important de décrire ce qui nous a amené à proposer une telle architecture. Une explication a été fournie dans le "rapport de faisabilité" joint en annexe (annexe n° 2) et nous y renvoyons pour des éclaircissements. Il fait suite au "cahier des charges" et constituent ensemble les deux premières étapes suivies pour la conception du logiciel, conformément à la méthode de travail propre à l'entreprise.

1 - Le module Interface Xybion

C'est ce module qui est chargé de gérer l'extraction des données du logiciel Xybion. Celui-ci contient toutes les données relatives aux animaux en phase Test ou Pretest des études réalisées depuis plusieurs années.

Le prélèvement des données de Xybion s'effectue par extraction selon le code d'une étude, pour un déterminant donné. Il concerne donc les données d'une seule étude mais permet de gérer un ou plusieurs déterminants à la fois, c'est-à-dire un ou plusieurs paramètres biologiques pour chaque déterminant.

Ce choix de conception provient du fait que la quantité de données à traiter peut être importante puisqu'elles concernent un ou plusieurs déterminants avec un ou plusieurs paramètres biologiques. De plus, pour des raisons de facilité offerte à l'utilisateur, l'extraction selon une étude semble préférable. Elle permet de traiter toute une étude en une seule fois et de scinder le travail.

Le résultat de l'extraction est mis dans une structure particulière assurant l'indépendance vis-à-vis de l'origine des données et du type d'extraction réalisé (cfr.module "Extraction-conversion" ci-dessous).

C'est aussi ce module qui est responsable du transfert des données extraites vers le module "Traitement des entrées". En effet, il faut permettre à l'utilisateur de compléter les informations provenant de Xybion ou de les mettre à jour. Cette fonction est assurée par ce dernier module.

Le module "Interface Xybion" s'occupe aussi du garnissage de la base de données avec l'ensemble des informations (de Xybion, plus celles qui ont été ajoutées), c'est-à-dire avec des résultats de la phase Prétest ou de Test, pour une étude donnée.

Il faut noter par ailleurs que ce module assure que les données transférées de Xybion vers la base de données l'ont été en totalité, correctement et de façon cohérente.

2 - Le module Interface Xybani

Les modalités de prélèvement des données sont les mêmes que pour le logiciel Xybion. Il a les mêmes fonctionnalités, si ce n'est qu'il concerne uniquement les données d'animaux en quarantaine et des non rongeurs. Elles proviennent du logiciel Xybani. Celui-ci ne contient aucune donnée concernant les rongeurs !

Ce module est également chargé du transfert des données extraites de Xybani vers le module "Traitement des entrées".

Il s'occupe enfin de la mise à jour ou garnissage de la base de données en ce qui concerne les résultats de quarantaine, pour une seule étude. Il assure que les données transférées sont correctes, cohérentes et qu'elles ont été transférées en totalité.

3 - Le module Interface Entrées manuelles

Il est chargé de gérer l'entrée des données autres que celles déjà stockées ailleurs. Ces entrées de données se font manuellement, par étude. Par exemple, une entrée manuelle peut être nécessaire dans le cas où on ne peut pas disposer des logiciels Xybion et Xybani, ou lors de pannes d'appareils de mesures. Cette dernière fonction est assurée par le module "Traitement des entrées".

Il assure le caractère cohérent et complet des données concernant une étude. Il s'occupe aussi du transfert de toutes ces informations vers la base de données (garnissage) en vérifiant qu'elles sont correctes, cohérentes et qu'elles ont été transférées en totalité.

4 - Le module Interface Entrées Poids

Ce module gère l'entrée des informations concernant les mesures de poids d'animaux au cours de leur vie.

Le traitement de ce type de données a été séparé afin de simplifier la tâche de l'utilisateur. Celui-ci peut entrer ses données de poids en une seule fois, quand il le souhaite, sans devoir passer par les interfaces manuelles, Xybion ou Xybani.

Le module "Interface Entrées poids" est aussi responsable du transfert de ces informations de poids vers la base de données. Il en assure le caractère cohérent et complet.

5 - Le module Interface Entrée Paramètre/déterminant

Il se charge de la gestion des paramètres ou déterminants susceptibles d'être considérés dans la base de données. Pour rappel, un déterminant (ou "détermination" selon la terminologie utilisée par Xybion) regroupe un ensemble de paramètres. Il ne peut exister sans le rattachement à au moins un de ceux-ci.

Actuellement la liste des paramètres utilisés lors d'études, est celle donnée en annexe (cfr.annexe n° 4). Comme elle peut être sujette à des modifications suscitées par l'apparition de nouvelles mesures ou de nouveaux critères, il est apparu nécessaire d'en prévoir la gestion.

C'est ainsi que l'utilisateur peut, quand il le veut, ajouter ou supprimer un paramètre biologique (se rapportant à un déterminant). Il doit pour cela le décrire complètement (nom du paramètre et unité de mesure associée). De même, il peut ajouter ou supprimer un déterminant.

Le module assure que toute modification de création ou de suppression s'effectue correctement et de façon cohérente. Tout déterminant doit en effet être relié à au moins un paramètre biologique et il est unique.
Tout paramètre biologique doit être rattaché à un déterminant au moins.

6 - Le module Conversion-Extraction

Ce module assure la récupération, sous forme de fichiers, des données présentes dans les logiciels Xybion et Xybani. Cela s'effectue selon une étude donnée et pour un seul déterminant à la fois.

L'extraction consiste en fait en une requête formulée dans le logiciel Xybion (ou Xybani). Elle a pour objet d'extraire (sous forme de fichier ASCII) les informations concernant l'étude et le déterminant qu'on a spécifié.

Pour des raisons pratiques (logiciel Xybion installé sur un autre "Account" et soumis à certains droits d'accès), il n'est pas possible d'effectuer cette extraction

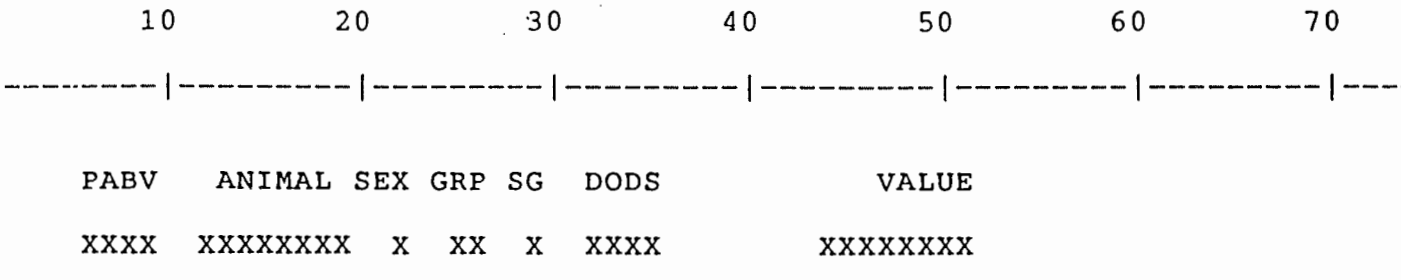


Figure 30 : Format d'une extraction de XYBION

directement à partir de notre logiciel.

Les informations obtenues se présentent sous la forme représentée à la figure 30 . On distingue que chaque élément a le format suivant : nom du paramètre, numéro d'animal, sexe, groupe, sous-groupe, jour d'étude, valeur du résultat. Il ne concerne qu'un seul paramètre biologique par extraction !

Nous avons donc décidé de permettre à l'utilisateur de traiter plusieurs paramètres biologiques à la fois et plusieurs déterminants. C'est pourquoi ce module assure l'extraction d'autant de données concernant des paramètres biologiques d'un même ou de plusieurs déterminants qu'on le désire.

En plus de l'extraction proprement dite, le module gère la conversion des structures obtenues par Xybion, en une structure de forme standard (FICH_CONVERTI).
Le choix d'une structure standard unique et qui nous est propre se justifie par le fait qu'elle permet de rassembler tout le contenu des extractions réalisées à partir de Xybion/Xybani.

Deux raisons nous ont amené à sa réalisation. D'abord, les structures de données provenant d'une extraction ne sont pas utilisables telles quelles; une conversion en des structures manipulables par Powerhouse doit être réalisée.

Ensuite, cela nous permet de rester indépendant de l'origine des données à considérer. Par exemple, dans le cas où l'on disposerait d'une nouvelle version de Xybion produisant un autre type de structure lors d'extractions, il suffit de prévoir une conversion de ce nouveau type de structure en notre format standard.

En ce qui concerne la structure FICH_CONVERTI, elle peut se représenter comme suit. Toute donnée est représentée sous la forme :

Numéro d'animal|sexe|nom déterminant|nom parametre|jour étude|valeur resultat

Ce sont les seules caractéristiques parmi celles contenues dans Xybion qu'il nous a été demandé de traiter pour la base de données. Elles peuvent concerner un ou plusieurs déterminants en même temps.

7 - Le module traitement des entrées

Il assure la gestion de toutes les saisies de données. Qu'il s'agisse des données complémentaires à celles extraites du logiciel Xybion ou Xybani, ou bien des données manuelles, entrées en fonction des besoins.

Il permet l'entrée des éléments communs à une étude, des éléments variables d'un animal à l'autre, ainsi que des éléments propres à chaque résultat, individuellement.

Le principe de ce module repose sur la présence de trois types distincts d'éléments pouvant être considérés comme des données :

- les éléments communs :

Ce sont les éléments (données) communs aux animaux de toute une étude. Ils sont à introduire par l'utilisateur, une seule fois, puisqu'ils caractérisent toute l'étude. Nous avons décidé de les regrouper dans une structure propre (ELEMCOMMUNS).

Les différents éléments qui peuvent être considérés comme communs à tous les animaux sont :

- le code de l'étude
- la date du début d'étude
- le type d'aliment
- l'espece
- la race
- la phase d'étude (quarantaine,pretest ou test)
- la voie d'injection
- le type d'échantillon
- l'état alimentaire
- le véhicule utilisé

- les éléments variables :

Ce sont les éléments (données) qui sont variables d'un animal à l'autre. Ils concernent les caractéristiques d'un animal. En effet, à un animal peut correspondre plusieurs entrées correspondant à des résultats observés pour cet animal. Il est dès lors inutile de devoir, pour chaque résultat, introduire les caractéristiques du même animal.

Nous avons décidé de les regrouper dans une structure propre (ELEMVARIABLES).Les différents éléments entrant dans cette catégorie des caractéristiques propres à chaque individu, sont :

- le numéro d'animal
- le sexe de l'animal
- l'âge

- la période d'acclimatation
- la date d'entrée dans l'animalerie
- la durée de la quarantaine
- le volume administre
- le nom du fournisseur
- le numéro d'animal en lot
- le code étude de quarantaine (pour les non rongeurs)

- les éléments hypervariables :

Ce sont les éléments (données) qui sont autres que les deux types d'éléments déjà cités. Ce sont les éléments qu'il faut introduire pour chaque résultat concernant un même animal.

Nous avons également décidé de les regrouper dans une structure propre (ELEMHYPERVARIABLES).
Les différents éléments entrant dans cette catégorie sont:

- la méthode d'analyse
- la date du résultat
- la valeur du résultat
- le nom du paramètre biologique
- le nom du déterminant
- le jour de l'étude
- le commentaire

Il est donc apparu nécessaire de traiter ces différents types d'éléments de façon distincte.

De plus, vu l'origine des données, il faut envisager différents types de saisie de données. En effet, les données extraites du logiciel Xybion constituent des éléments variables aussi bien que des éléments hypervariables selon ce qui a été exposé ci-dessus, et sont déjà disponibles. Il ne faut donc plus les saisir à l'écran.
Ceci est aussi vrai pour le logiciel Xybani.
Pour les entrées manuelles, aucune donnée n'est disponible au départ de la saisie proprement dite.

Ce module permet enfin d'assurer le caractère complet et cohérent des données introduites lors de la saisie à l'écran. Des contrôles divers quant à la syntaxe et au contenu sont envisagés ici.

Ce module de saisie et traitements des entrées sera typiquement réalisé sous forme d'écran Powerhouse (cfr.chap.3.2). Chaque type d'entrée (élément commun, variable ou hypervariable) donnera lieu à un sous-module permettant la saisie des données correspondantes.

8 - Le module transfert

Ce module est chargé du transfert des données contenues dans la structure FICH_CONVERTI (cfr.module "Conversion-Extraction") qui contient l'ensemble des informations extraites de Xybion/Xybani, vers le module traitement des entrées.

Ce transfert s'effectue selon le principe de l'existence d'éléments communs, variables et hypervariables comme décrit ci-dessus (cfr.module "Traitement des entrées"). De plus il distingue le transfert de données de la phase Test des transferts des données de la phase Prétest et de Quarantaine. En effet, le type de traitement à effectuer diffère selon les cas.

9 - Le module Garnissage

Il est chargé de garnir la base de données à partir des informations fournies lors des entrées et concernant une étude particulière.

Il assure le caractère cohérent et complet des données stockées dans la base de données. Il établit pour cela tous les contrôles nécessaires (contrôles d'intégrité et de redondance) en accédant aux données de la base de données elles-mêmes.

En ce qui concerne le caractère cohérent, le module se base pour le garnissage sur les principes suivants.

Toute donnée concernant un animal peut être identifiée à partir du numéro identifiant cet animal. Nous avons pris comme convention de stocker toutes les données avec un identifiant (num_animal) constitué par la fusion du numéro de quarantaine d'un animal avec son code étude définitif. En effet, cette convention permet un minimum de mises à jour des identifiants dans la base de données.

Typiquement, trois situations peuvent se présenter lors du garnissage de la B.D. :

- le garnissage avec des données de quarantaine :

c'est-à-dire des données de non rongeurs uniquement car il n'existe pas de données de quarantaine pour les rongeurs !

Dans ce cas, si les données concernant les mêmes animaux ont déjà été stockées dans la base de données sous la forme (num_animal = nr_lot + code_def), alors il faut enregistrer les données de quarantaine sous ce même num_animal.

Par contre, si aucune donnée n'existe déjà pour ces animaux, il faut stocker les données de quarantaine sous la forme (num_animal = nr_lot + code_etude_bidon).

- le garnissage avec des données de prétest :

c'est-à-dire concernant aussi bien les rongeurs que les non rongeurs !
Dans ce cas,

* si des données concernant ces mêmes animaux ont déjà été stockées dans la base de données sous la forme (num_animal = nr_lot + code_def), alors il faut enregistrer les données de prétest directement avec ce même num_animal.

* si aucune donnée n'existe déjà, il faut stocker les données de prétest sous la forme (num_animal = nr_lot + code_def).

* si des données concernant ces mêmes animaux ont déjà été stockées dans la base de données sous la forme (num_animal = nr_lot + code_etude_bidon) et s'il s'agit de rongeurs, il faut enregistrer les données de prétest avec le numéro (num_animal = nr_lot + code_def) et convertir les données déjà présentes pour leur attribuer ce même num_animal.

- le garnissage de données de test:

c'est-à-dire concernant aussi bien les rongeurs que les non rongeurs.
Dans ce cas, on peut directement enregistrer les données de Test sous la forme num_animal = nr_lot + code_def. Mais en plus, il faut convertir le num_animal des données déjà stockées qui pourraient exister sous la forme (num_animal = nr_lot + code_etude_bidon) pour des non rongeurs, en num_animal = nr_lot + code_def.

Lors de la conception, il a été envisagé de créer des tables de référence ou "cross-list" garnies chaque fois qu'il y a de nouvelles informations disponibles concernant les numéros ou codes études des animaux. La mise à jour de ces tables se faisant par le module garnissage lui-même.
On distingue alors la table répertoriant l'équivalence entre les numéros de lot (nr_lot) et les numéros définitifs (nr_def) de chaque animal (table CL_NUMERO). La table CL_CODE donne l'équivalence entre le code quarantaine (code_etude_bidon) et le code étude définitif (code_def).

Ces tables sont constamment accédées lors du garnissage afin de contrôler la forme sous laquelle sont stockées les données déjà présentes ou pour garnir la B.D. avec l'identifiant (num_animal) dont on dispose à ce moment.

10 - Le module accès B.D.

Il est chargé de gérer tous les accès à la base de données. il en contrôle la sécurité d'accès.

11 - Le module Création

Ce module gère toutes les créations de données nécessaires dans la base de données. Il en assure le caractère cohérent.

12 - Le module Suppression

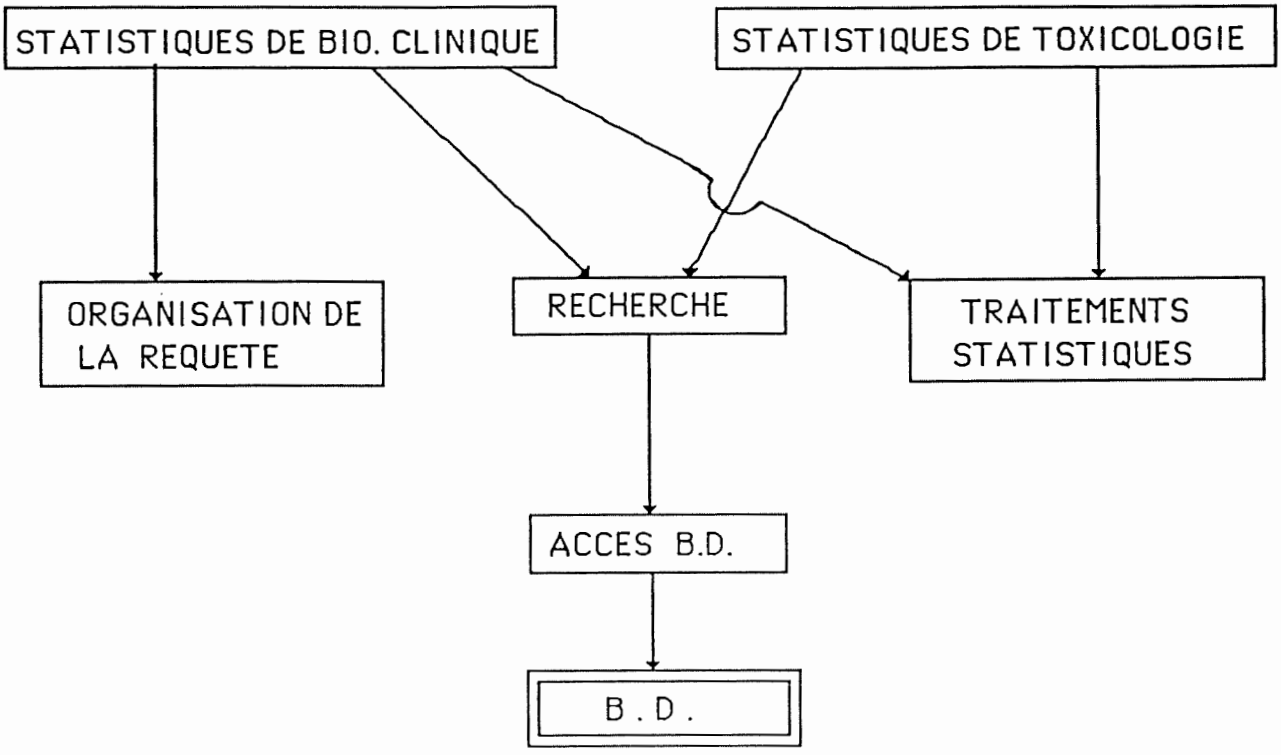
Ce module gère toutes les suppressions de données nécessaires dans la base de données. Il en assure le caractère cohérent.

13 - Le module mise-à-jour

Ce module gère toutes les mises-à-jour de données dans la base de données. Il en assure le caractère cohérent.

B. SECONDE PARTIE : EXPLOITATION DE LA B.D.

Voici l'architecture logique de cette partie du projet :



- à la recherche de l'information dans la B.D. sur base de critères bien spécifiques

- au traitement de l'information recueillie en y appliquant les statistiques demandées

Voyons maintenant d'un peu plus près le principe de chaque module.

1. Module "Accès B.D."

Il a déjà été commenté dans le point 4.4.3.A .

2. Module "Organisation de la requête"

Ce module est propre à la partie biologie clinique et n'est donc pas utilisé par le module "Statistiques de toxico".

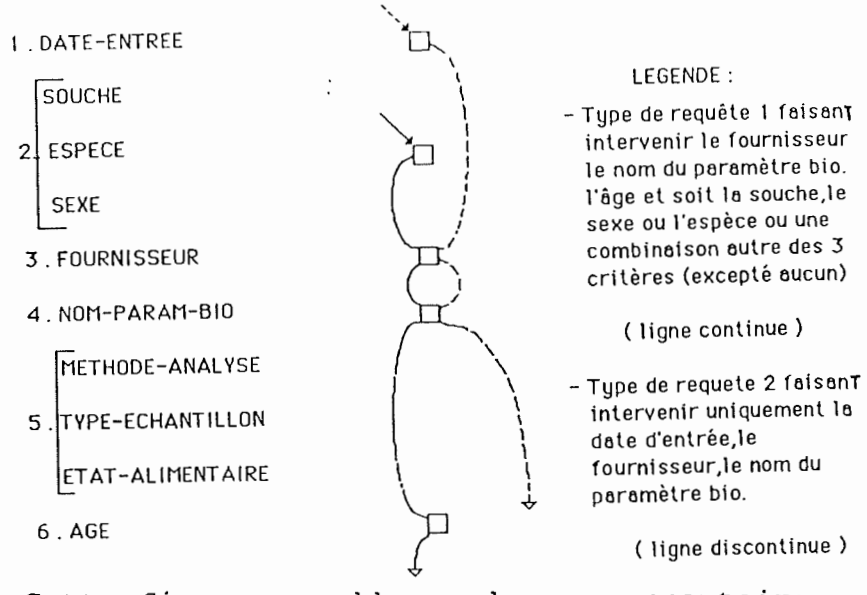
* But : à partir des critères entrés par l'utilisateur et qui comptent parmi :

- la date d'entrée dans l'animalerie
- l'espèce
- la race
- le sexe
- le fournisseur
- le nom du paramètre biologique
- la méthode d'analyse
- le type d'échantillon
- l'état alimentaire
- l'âge au moment de la mesure,

le module établit la structure de la requête en terme des modules de recherche qui seront effectivement utilisés.

* Principe :

Dans la plupart des cas , tous les critères proposés ne seront pas pris en compte par les utilisateurs. Seulement un certain nombre seront spécifiés. Il faut dès lors éviter d'effectuer des opérations blanches en se basant sur un critère non rempli. La figure ci-dessous illustre le problème en spécifiant les dix critères et quelques types de requête qui apparaissent sous forme de chemins (voir légende).



Cette figure appelle quelques commentaires :

+ chaque carré représente un sous-module du module de recherche qui sélectionne des informations sur les critères qui s'y rattachent

+ certains critères ont été regroupés pour former un seul sous-module. Par exemple , les critères "Espèce" , "Souche" et "Sexe" forment un seul sous-module. La raison de ce regroupement (ainsi que des trois critères "Méthode analyse" , "Type échantillon" , "Etat alimentaire") est qu'ils appartiennent au même concept de la B.D. Si l'un ou l'autre de ces critères n'étaient pas spécifié , il suffirait de ne pas en tenir compte dans l'obtention des données.

+ les flèches indiquent le premier critère spécifié et la fin de la recherche.

+ les deux exemples de type de requête passent d'office par le critère "Nom du paramètre bio". Celui-ci est en effet exigé à l'utilisateur sans quoi cela n'aurait plus aucun sens d'effectuer la recherche !

A priori , il n'est pas possible de connaître le type de requête souhaité par l'utilisateur. Ce module sera responsable d'identifier précisément les opérations à réaliser. Une solution aurait été de concevoir un module pour chaque type de requête comportant tous les accès nécessaires. Par exemple , la demande pour le paramètre biologique "glucose" , pour des animaux de 5 ans et pour l'espèce "singe" constituerait un module type (recherche selon l'ESPECE , le PARAMETRE BIOLOGIQUE , l'AGE).

Cette solution a été délaissée au profit d'une solution où un module existe pour chaque critère et groupe de critères (cfr "Espece","Race","Sexe"). Cette optique permet de diminuer le nombre de modules à prévoir et les rend plus simples. Il est à noter que les données sélectionnées pour chaque critère (ou groupe de critères) seront placés dans des structures intermédiaires particulières à savoir :

1. critère	Date-entrée	FRUIT-SELECTION-1
2. critères	Espèce Souche Sexe	FRUIT-SELECTION-2
3. critère	Fournisseur	FRUIT-SELECTION-3
4. critère	Nom-paramètre-bio	FRUIT-SELECTION-4
5. critères	Methode-analyse Type-échantillon Etat-alimentaire	FRUIT-SELECTION-5
6. critère	Age	FRUIT-SELECTION-6

En réalité , il y aura plus de six sous-modules de recherche. Il est clair qu'avant de sélectionner des données sur base d'un critère , des premières sélections ont déjà pu être réalisées sur base d'un autre critère (ainsi en sélectionnant sur base du "Nom-paramètre-bio" , une sélection a dèjà pu être réalisée selon l'espèce et dans ce cas bien précis , il faut se baser sur les données présentes dans FRUIT-SELECTION-2 pour continuer le processus de recherche). Il faudra donc discerner les diverses origines précedant un sous-module donné : soit il s'agit du premier critère et on se situe à un point d'entrée , soit on a déjà réalisé des sélections suivant d'autres critères et il faut alors se baser sur les informations déjà recueillies (présentes dans FRUIT-SELECTION-1,2,3,4 ou 5) comme base de la sélection à venir.

Au lieu de créer les structures intermédiaires mentionnées ci-dessus , toutes les données intermédiaires auraient pu être placées dans une structure avec un format commun. Cette solution présentait l'avantage de la simplicité mais l'inconvénient d'une performance moindre. Il aurait , en effet , fallu prévoir un grand nombre d'articles et parmi ceux-ci , peu auraient été garnis ,ce qui aurait conduit à une taille exagérément grande de la structure et une baisse de performances.

Toutes les données provenant d'un processus de recherche sur l' ensemble des critères proposés à l'utilisateur doivent finalement figurer dans la structure de données FRUIT-SELECTION-6 ayant le format :

+ valeur du résultat

+ état (phase de l'étude)

+ numéro d'animal (num-animal)

+ numéro de condition expérimentale

+ date du résultat

Il faut donc que si la recherche s'arrête au critère "Nom-paramètre-bio" ou à un des trois critères "Méthode-analyse" , "Type-échantillon" , "Etat-alimentaire" , les données se trouvent sous ce format. Le choix d'un tel format final unique a été décidé pour une raison de simplicité.

L'ordre des critères qui est donné juste avant ne correspond pas à l'ordre qui a été avancé au point 4.3.2 paragraphe 1 et qui est respecté dans la demande à l'utilisateur . C'est délibérément que ce choix a été fait

dans le but d'optimiser le parcours au sein de la B.D.

Pour plus de précision concernant les arguments reçus et à passer , on peut se référer au paragraphe 4.5.2 (1.) de la conception physique du logiciel.

3. Module "Recherche"

* But : il a pour objectif de répondre à toute une série de demandes de recherche .

* Liste des demandes à assurer :

- Liste des animaux entrés à une date déterminée dans l'animalerie
- Liste des animaux entrés avant une date déterminée dans l'animalerie
- Liste des animaux entrés après une date déterminée dans l'animalerie
- Liste des animaux de telle espèce
- Liste des animaux de tel sexe
- Liste des animaux de telle race
- Liste des animaux ayant été livré par un fournisseur déterminé
- Liste des résultats correspondant à un paramètre biologique donné
- Liste des résultats correspondant à une méthode d'analyse donnée
- Liste des résultats correspondant à un type d'échantillon donné
- Liste des résultats correspondant à un état alimentaire donné
- Liste des résultats pour des animaux d'un âge donné
- Liste des résultats pour des animaux d' âge inférieur à un âge donné
- Liste des résultats pour des animaux d' âge supérieur à un âge donné
- Obtention des âges des animaux dont les résultats ont

été sélectionnés suite à une recherche selon les critères de biologie clinique

- Obtention des poids des animaux dont les résultats ont été sélectionnés suite à une recherche selon les critères de biologie clinique

- Obtention des types d'aliment des animaux dont les résultats ont été sélectionnés suite à une recherche selon les critères de biologie clinique

- Obtention des fournisseurs des animaux dont les résultats ont été sélectionnés suite à une recherche selon les critères de biologie clinique

- Obtention des races des animaux dont les résultats ont été sélectionnés suite à une recherche selon les critères de biologie clinique

- Obtention des espèces des animaux dont les résultats ont été sélectionnés suite à une recherche selon les critères de biologie clinique

- Obtention des types d'injection des animaux dont les résultats ont été sélectionnés suite à une recherche selon les critères de biologie clinique

- Liste des poids des animaux pour chaque jour d'une étude

- Liste des poids des animaux pour chaque jour d'une étude par sexe

- Liste des poids des animaux en fonction de l'âge pour une espèce , une race et un sexe déterminés

- Liste des poids des animaux en fonction de l'âge pour une espèce et un sexe déterminés

- Liste des poids des animaux en fonction de l'âge pour une espèce et une race déterminées

Ce module apparaît donc plutôt comme une librairie de sous-modules réalisant chacun une requête bien établie.

Chacun de ceux-ci est explicité au paragraphe 4.5.1 B. (2.) de l'Architecture du logiciel.

4. Module "Statistiques de biologie clinique"

* But : il joue le rôle d'interface avec l'utilisateur pour les statistiques de biologie clinique.

* Caractéristiques fonctionnelles : il assure un certain nombre de fonctions.

a. c'est ce module qui propose les dix critères de sélection pour une recherche de bio. clinique. Ceux-ci sont proposés à l'utilisateur suivant l'ordre indiqué au point 4.3.2 par. 1 . Quelque soit le type de statistiques demandé par l'utilisateur , il devra passer par ces critères qui constituent la première étape obligatoire.

Les critères spécifiés seront placés dans la strucure REQUETE-BIOL qui comporte outre ces dix critères :

- l'opérateur pour l'âge (operateur-age) : l'utilisateur pourra entrer trois types de recherche sur l'âge;
  - + demander les résultats pour des animaux d'un âge précis (operateur-age vaut " = ")
  - + demander les résultats pour des animaux ayant plus d'un âge donné (operateur-age vaut " > ")
  - + demander les résultats pour des animaux ayant moins d'un âge donné (operateur-age vaut " < ")
- l'opérateur de la date d'entrée (operateur-date) ; le principe est analogue à l'opérateur pour l'âge
- l'unité de mesure du paramètre biologique entré (unite-mesure)
- le type de statistique (typestat) demandé (voir point b ci-après). Le but de cet élément est de permettre au module "Traitements statistiques" de déterminer quel traitement doit être effectué . Une liste des abbréviations est donnée à l'annexe n° 3 .

b. le choix du type de statistique intervient à ce niveau et constitue une étape de préparation des données obtenues par la recherche pour les traitements statistiques à venir. La liste proposée est celle obtenue lors de l'"Analyse de besoins" et figure au point 4.3.2 paragraphe 2 sous-par "Biologie clinique".

c. "Statistiques de biologie clinique" utilise :

- le module "Organisation de la requête"
- le module "Recherche"
- le module "Traitements statistiques"

5. Module "Statistiques de toxico"

* But : il joue le rôle d'interface avec l'utilisateur pour les statistiques de toxicologie.

* Caractéristiques fonctionnelles :

a. les critères de la toxicologie sont présentés , à savoir :

- code de l'étude
- espèce
- race
- sexe

Il est à remarquer que les critères proposés ici , contrairement à ceux proposés dans la partie Biologie clinique , ne sont pas les mêmes à chaque fois. Ils dépendent du type de statistique qui sera demandé.

b. les statistiques offertes sont présentées à ce niveau , et comme il a été dit ci-dessus , les critères proposés sont variables. La liste proposée est celle obtenue lors de l'"Analyse des besoins" et figure au point 4.3.2 par. 2 sous-par. "Toxicologie".

c. "Statistiques de toxico" utilise :

- le module "Recherche"
- le module "Traitements statistiques"

6. Module "Traitements statistiques"

* But : il est responsable de tous les traitements statistiques à effectuer que ce soit de biologie clinique ou de toxicologie. Il est donc responsable de l'orientation vers la demande désirée.

* Fonctions : tous les traitements demandés ont été évoqués au paragraphe 4.3.2 par. 2 . "Traitements statistiques" sera donc plutôt un ensemble de petits modules répondant aux requêtes statistiques avec en plus un module directeur qui orientera vers le traitement adéquat. Une description détaillée sera donnée au paragraphe 4.5.1 B.(5.) de la conception physique du logiciel.

4.5. CONCEPTION PHYSIQUE.

La méthode de conception adoptée sur place n'attribue pas une importance fondamentale aux spécifications de chaque module.

Il semble quand même intéressant, surtout en perspective d'une maintenance possible à effectuer sur le logiciel par d'autres personnes, de préciser les fonctionnalités précises de chaque module physique.

C'est pourquoi nous décrivons ci-après chaque module physique de notre architecture. Dans un premier temps, nous nous intéresserons aux modules intervenant dans la conception de la base de données. ensuite seront spécifiés les modules propres à l'exploitation de la B.D.

4.5.1. SPECIFICATION DES MODULES.

La spécification des fonctionnalités de chaque module est réalisée de la façon suivante.
Pour tout module, on spécifie les données (D.) à fournir en entrée de ce module et qui lui sont nécessaires pour son bon déroulement.

On spécifie ensuite les conditions que doivent respecter ces données en entrée (C.D). C'est-à-dire les conditions qui doivent être satisfaites pour assurer le bon fonctionnement du module.

Ensuite, sont précisés les résultats produits par le module (R.).

Ils sont suivis des conditions portant sur les résultats produits (C.R). Selon les cas, une liste des modules appelés par le module considéré est jointe à la fin de ces descriptions.

PARTIE A : Constitution de la B.D.

Le module ENTREES_XYBION

D.: Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES, CL_NUMERO, PRETEST, TAB_CHANGEMENTS, FICH_CONVERTI, les fichiers d'extraction de Xybion et les fichiers de la base de données.

C.D.: - La structure des fichiers extractions est celle donnée au point 4.3.2 lors de la description du module "Conversion-extraction".

- La structure du fichier FICH_CONVERTI est la suivante. C'est un fichier de forme standard dont

chaque record a la structure:

NR_ANIMAL SEXE NOMDET NOMPARAM JOURETUDE VALEUR

- Le fichier CL_NUMERO contient des records de structure :

NR_LOT NR_DEF

Il contient les correspondances, pour tout animal, entre son numéro de lot et son numéro définitif.

- le fichier TAB_CHANGEMENTS contient des records de structure :

NR_LOT NR_DEF CODE_BIDON CODE_DEF

Il contient les informations d'identification dont on dispose pour un animal avant de lancer un garnissage. Pour les rongeurs, il ne peut exister de valeur pour le CODE_BIDON.

- les fichiers de la base de données ont la structure présentée en annexe (cfr.annexe n° 7).

- le fichier ELEMCOMMUNS contient des records dont les champs sont : le code étude (prégarni)
  la date du début d'étude (prégarni)
  le type d'aliment (obligatoire)
  l'espèce (obligatoire)
  la race (obligatoire)
  l'etat ou phase (prégarni)
  la voie d'injection (obligatoire)
  le type d'échantillon (obligatoire)
  l'état alimentaire (obligatoire)
  le véhicule utilisé (obligatoire)

  Les champs "prégarnis" proviennent des valeurs extraites de Xybion et qu'il ne faut donc plus compléter. En ce qui concerne le caractère "obligatoire" des champs, nous avons adopté un système de vérification des valeurs entrées en fonction de tables reprenant l'ensemble des différentes valeurs pouvant exister pour chaque champs.
  Pour détourner ce caractère obligatoire (qui semble cependant une bonne guarantie pour la complétude des données), l'utilisateur peut définir l'existence de la valeur "blanche" dans ces tables contenant l'ensemble des valeurs permises (cfr.module "MAJ_TABLES").

- le fichier ELEMVARIABLES contient des records dont les champs sont :
  le num_animal (obligatoire,automatique)
  le nr_animal (obligatoire)
  le sexe (prégarni)
  la durée quarantaine (obligatoire)

l'âge de l'animal (obligatoire)
la période d'acclimatation (obligatoire)
la date d'entrée (obligatoire)
le nom du fournisseur (obligatoire)
le numéro de fournisseur (automatique, obligatoire)
le volume administré (obligatoire)
le code étude de quarantaine (parfois obligatoire)
le numéro de lot (parfois obligatoire)

Les champs "obligatoires automatiques" sont des champs internes construits à partir des autres champs. Le num_animal est constitué du numéro d'animal + le code de l'étude. Le num_fournisseur est un numéro attribué séquentiellement à tout nouveau fournisseur ou bien le numéro du fournisseur correspondant au nom introduit.

De plus, il faut signaler que le fichier NUM_MAX contient à tout moment, le dernier numéro de fournisseur attribué. En l'accédant, il permet de trouver directement le numéro à attribuer à un nouveau fournisseur.

Les champs "parfois obligatoires" proviennent du fait que

+ le code étude quarantaine n'est nécessaire que pour des données Test et des animaux non rongeurs. En effet, pour des données de quarantaine, c'est le code étude introduit dans le fichier ELEMCOMMUNS qui constitue ce code étude quarantaine.

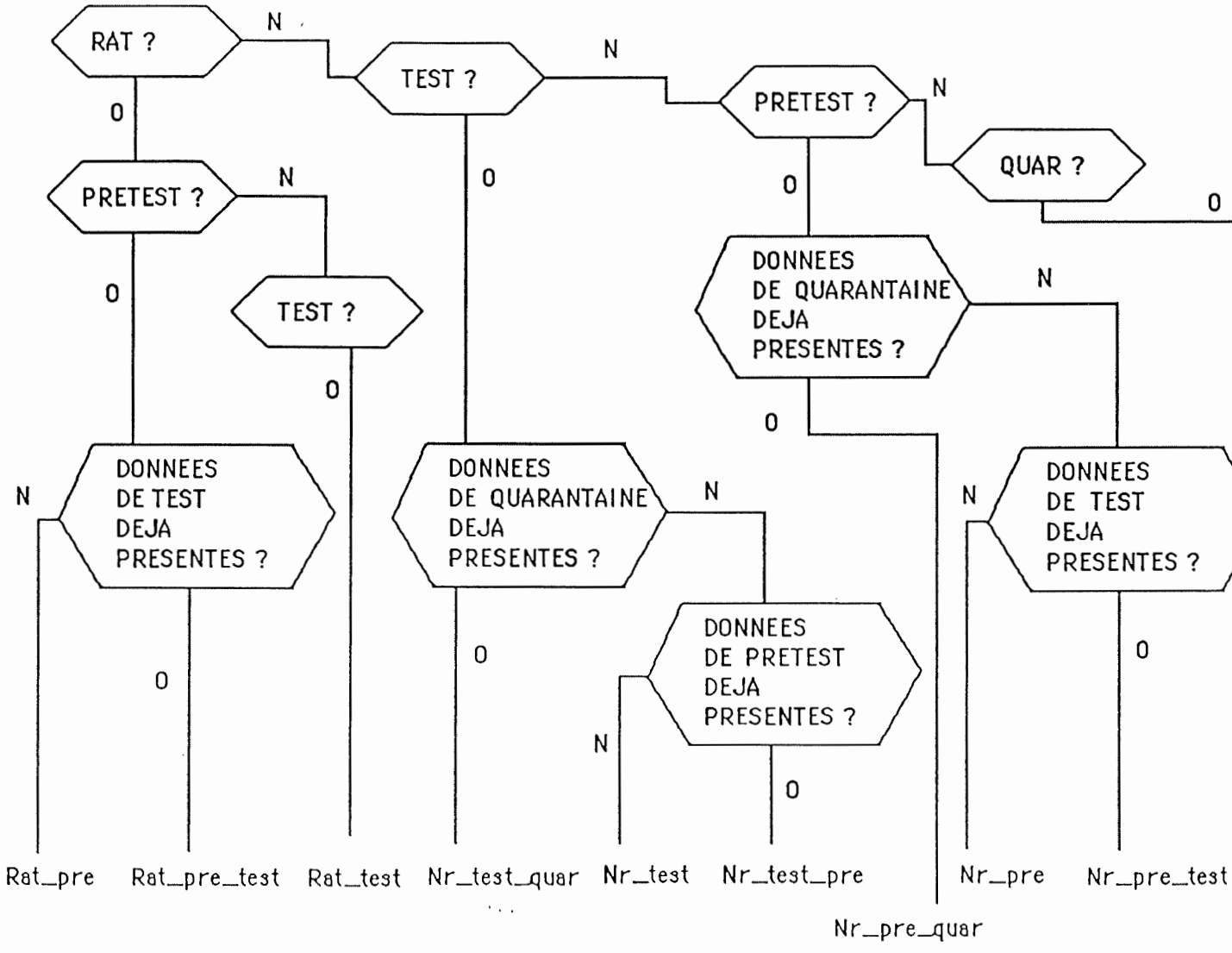
+ le numéro de lot n'est nécessaire que pour des données TEST. En effet, pour le cas de données PRETEST, c'est le numéro d'animal déjà présent dans ELEMVARIABLES !

- le fichier ELEMHYPERVARIABLES contient des records dont les champs sont :

le num_animal (automatique, obligatoire)
le nom de déterminant (prégarni)
le nom du paramètre biologique (prégarni)
la méthode d'analyse (obligatoire)
la date du résultat (prégarni)
le jour d'étude (prégarni)
la valeur du résultat (prégarni)
le commentaire (facultatif)

R.: les données extraites du logiciel Xybion concernant les animaux en phase Test ou Pretest (il n'est pas possible

Figure 31 : DIFFERENTS TYPES DE GARNISSAGE



GARNISSAGES XYBANI : Garnissage QUAR uniquement

GARNISSAGE XYBION : Tous les garnissages sauf QUAR

GARNISSAGE MANUEL : Tous les garnissages

ici de considérer les deux types de donnée à la fois), pour une étude donnée, sont :

- soit dans les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES
- soit dans les fichiers de la base de données
- soit dans les fichiers d'extraction de Xybion
- soit dans le fichier FICH_CONVERTI

La procédure suivie consiste à établir la conversion des fichiers extractions en un fichier FICH_CONVERTI standard, de transférer des données Prétest de ce fichier vers le module traitement des entrées, de permettre de compléter les informations concernant la phase Pretest , de transférer les données de la phase Test, de permettre de compléter les informations concernant la phase Test et enfin de permettre le garnissage de la base des données.

C.R.:

- si seul le transfert a été réalisé, les données restent dans les trois fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES. Elles y restent même pour une autre session d'utilisation. Il est toujours possible de les modifier. A un moment déterminé, il ne peut cependant exister que des données de phase Prétest ou bien de phase TEST mais pas les deux à la fois.

D'autre part, au niveau du transfert, des contrôles sont effectués pour s'assurer que les fichiers de destination (ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES) sont vides. Ceci empêche de transférer d'autres données si on n'a pas encore garni la base de données avec celles dont on dispose déjà !

- si seul le garnissage a été fait, les données sont dans la base de données. Les trois fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES sont vides. Le fichier TAB_CHANGEMENT est vide.

A ce niveau plusieurs contrôles sont effectués:

* il faut que les données pour le garnissage soient complètes du point de vue de l'utilisateur; concrètement, cela se vérifie en regardant si celui-ci a entré des informations complémentaires pour le fichier ELEMCOMMUNS (c'est-à-dire qu'il a, par exemple, spécifié au moins l'espèce) avant d'effectuer le garnissage. De même pour les fichiers ELEMVARIABLES (le volume administre est au moins nécessaire) et ELEMHYPERVARIABLES (la méthode d'analyse est au moins nécessaire).

* des garnissages spécifiques sont à réaliser en fonction du type de donnée (cfr.fig.31). En

effet, avant de garnir, on vérifie s'il n'existe pas déjà des données stockées dans la B.D. concernant cette étude et on regarde sous quels identifiants elles l'auraient été. En fonction de ceux-ci, on réalise un garnissage cohérent avec ce qui existe déjà.

Du point de vue pratique, il n'existe pas tous les différents types de garnissage qui apparaissent sur la figure car on remarque que du point de vue réalisation, certains garnissages sont rigoureusement identiques.
Ainsi, les quatres types de garnissage suivants sont identiques.

GARNISSAGE_RAT_PRE_TEST
GARNISSAGE_RAT_PRE
GARNISSAGE_NR_PRE_TEST
GARNISSAGE_QUAR

Nous les avons regroupé sous le nom GARNISSAGE_QUAR.
De même,

GARNISSAGE_NR_PRE est identique à GARNISSAGE_NR_PRE_QUAR . C'est le nom de ce dernier qui est utilisé pour les deux.

GARNISSAGE_NR_TEST_PRE est identique à GARNISSAGE_RAT_TEST. C'est le nom de ce dernier qui est utilisé pour les deux.

GARNISSAGE_NR_TEST est identique à GARNISSAGE_NR_TEST_QUAR. C'est le nom de ce dernier qui est utilisé pour les deux.

Il n'y a donc réellement que quatre types différents de garnissage à réaliser.

- si seule la conversion a été faite, les données restent dans le fichier FICH_CONVERTI. Il suffit à l'utilisateur, lors d'une prochaine session, de transférer son contenu vers les trois fichiers pour en disposer à nouveau.

- si rien n'a été fait, le module ne produit aucun résultat. Les données sont toujours dans les fichiers extractions de Xybion. Ceci correspond au retour au menu Powerhouse supérieur sans avoir choisi aucune option.

APPELS : ce module appelle les modules CONV_XYBION, TRANSFERT_TEST, TRANSFERT_PRETEST, ELEMENTS_COMMUNS_ION, GARNISSAGE_QUAR, GARNISSAGE_RAT_TEST, GARNISSAGE_NR_PRE_QUAR, GARNISSAGE_NR_TEST_QUAR.

Le module ENTREES_XYBANI

D.: Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES, FICH_CONVERTI, les fichiers d'extraction de Xybion et les fichiers de la base de données.

C.D.: - La structure des fichiers extractions est celle donnée au point 4.3.2 lors de la description du module "Conversion-extraction".

- La structure du fichier FICH_CONVERTI est celle donnée dans le module "ENTREES_XYBION".

- les structures des fichiers de la base de données sont celles présentées en annexe (cfr.annexe n° 7).

- le fichier ELEMCOMMUNS a la structure décrite dans le module "ENTREES_XYBION".

- le fichier ELEMVARIABLES a la structure décrite dans le module "ENTREES_XYBION".

- le fichier ELEMHYPERVARIABLES a la structure décrite dans le module "ENTREES_XYBION".

R.: les données extraites du logiciel Xybani concernant les animaux en phase quarantaine , pour une étude donnée, sont : -soit dans les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES
- soit dans les fichiers de la base de données
- soit dans les fichiers d'extraction de Xybion
- soit dans le fichier FICH_CONVERTI

La procédure suivie consiste à établir la conversion des fichiers extractions en un fichier FICH_CONVERTI standard de transférer les données (de quarantaine) de ce fichier vers le module traitement des entrées, de permettre de compléter les informations concernant la phase quarantaine et de permettre le garnissage de la base de données.

C.R.:

- si seuls la conversion et le transfert ont été réalisés, les données restent dans les trois fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES. Elles y restent même pour une autre session d'utilisation. Il est toujours possible de les modifier.

D'autre part, au niveau du transfert, des contrôles sont effectués pour s'assurer que les fichiers de destination (ELEMCOMMUNS, ELEMVARIABLES et

ELEMHYPERVARIABLES) sont vides. Ceci empêche de transférer d'autres données si on n'a pas encore garni la base de données avec celles dont on dispose déjà !

- si seul le garnissage a été fait, les données sont dans la base de données. Les trois fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES sont vides.

A ce niveau plusieurs contrôles sont effectués:

* il faut que les données pour le garnissage soient complètes du point de vue de l'utilisateur; concrètement, cela se vérifie en regardant si celui-ci a entré des informations complémentaires pour le fichier ELEMCOMMUNS (c'est-à-dire qu'il a, par exemple, spécifié au moins l'espèce) avant d'effectuer le garnissage. De même pour les fichiers ELEMVARIABLES (le volume administré est au moins nécessaire) et ELEMHYPERVARIABLES (la méthode d'analyse est au moins nécessaire).

* le seul garnissage spécifique à réaliser dans ici , est GARNISSAGE_QUAR (cfr.fig.31).

- si rien n'a été fait, le module ne produit aucun résultat. Les données sont toujours dans les fichiers extractions de Xybani. Ceci correspond au retour au menu Powerhouse supérieur sans avoir choisi aucune option.

APPELS : ce module appelle les modules CONV_XYBION, TRANSFERT_ANI, ELEMENTS_COMMUNS_ANI, GARNISSAGE_QUAR.

Le module ENTREES_MANUELLES

D.: Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES, CL_NUMERO, PRETEST, TAB_CHANGEMENTS, FICH_CONVERTI, les fichiers de la base de données.

C.D.: - La structure du fichier FICH_CONVERTI est celle décrite dans le module "ENTREES_XYBION".

- Le fichier CL_NUMERO est celui décrit dans le module "ENTREES_XYBION".

- le fichier TAB_CHANGEMENTS est celui décrit dans le module "ENTREES_XYBION".

- les fichiers de la base de données ont la structure présentée en annexe (cfr.annexe n° 7).

- les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES ont les structures décrites dans le module "ENTREES_XYBION".

R.: les données entrées manuellement concernant les animaux en phase Test, Pretest ou quarantaine (il n'est pas possible de considérer les trois types de données à la fois ici), pour une étude donnée, sont :

-soit dans les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES
- soit dans les fichiers de la base de données

La procédure suivie consiste à permettre d'entrer les données manuellement et de permettre le garnissage de la base des données.

C.R.:

- si seule l'entrée des données a été réalisée, les données restent dans les trois fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES. Elles y restent même pour une autre session d'utilisation. Il est toujours possible de les modifier. A un moment déterminé, il ne peut cependant exister que des données de phase Prétest ou bien de phase TEST ou bien de phase quarantaine, mais pas les trois à la fois.

D'autre part, au niveau du transfert, des contrôles sont effectués pour s'assurer que les fichiers de destination (ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES) sont vides. Ceci empêche de transférer d'autres données si on n'a pas encore garni la base de données avec celles dont on dispose déjà !

- si seul le garnissage a été fait, les données sont dans la base de données. Les trois fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES sont vides. Le fichier TAB_CHANGEMENT est vide.

A ce niveau plusieurs contrôles sont effectués:

* il faut que les données pour le garnissage soient complètes du point de vue de l'utilisateur; concrètement, cela se vérifie en regardant si celui-ci a entré des informations complémentaires pour le fichier ELEMCOMMUNS (c'est-à-dire qu'il a, par exemple, spécifié au moins l'espèce) avant d'effectuer le garnissage. De même pour les fichiers ELEMVARIABLES (le volume administré est au moins nécessaire) et

ELEMHYPERVARIABLES (la méthode d'analyse est au moins nécessaire).

* des garnissages spécifiques sont à réaliser en fonction du type de données (cfr.fig.31). En effet, avant de garnir, on vérifie s'il n'existe pas déjà des données stockées dans la B.D. concernant cette étude et on regarde sous quels identifiants elles l'auraient été. En fonction de ceux-ci, on réalise un garnissage cohérent avec ce qui existe déjà.
Du point de vue pratique, il n'existe pas tous ces différents types de garnissage (cfr. module "ENTREES_XYBION").

APPELS : ce module appelle les modules ELEMENTS_COMMUNS_MAN, GARNISSAGE_QUAR, GARNISSAGE_RAT_TEST, GARNISSAGE_NR_PRE_QUAR, GARNISSAGE_NR_TEST_QUAR.

Le module ENTREES_DONNEES_POIDS

D.: Les fichiers POIDS, ESPECE, POIDSINTER, TABLE_MESURE, CL_CODE, CL_NUMERO.

C.D.: - Le fichier TABLE_MESURE a la structure suivante :

ENTREE	FACTEUR	SORTIE
unité	facteur multiplicatif	unité transformée
Ex. kg	* 0.01	gr

Il permet de convertir automatiquement les valeurs de poids entrées, en valeurs standards pour le stockage dans la base de données. Les données de poids sont, par exemple, toujours stockées en gramme. Il permet aussi, dans le sens contraire, de convertir les unités de stockage dans la B.D. en d'autres unités, en fonction des besoins d'affichage à l'écran.

- le fichier CL_CODE a la structure suivante :

CODE_BIDON	CODE_DEF

Il contient les correspondances, pour tout animal non rongeur, entre son code étude bidon attribué pour les données de quarantaine et son numéro définitif une fois celui-ci connu.

- Le fichier CL_NUMERO a la structure décrite dans le module "ENTREES_XYBION".

- Les fichiers POIDS, ESPECE et POIDSINTER ont leur structure décrite en annexe (cfr annexe n° 7). POIDSINTER est un fichier de travail pour stocker temporairement les valeurs de poids. Il a la même structure que le fichier POIDS.

R.: Après saisie des valeurs de poids, le fichier POIDS reprend l'ensemble des mesures effectuées. Ce fichier contient les éléments suivants :

DATE_MESURE
VALEUR_POIDS
NUM_ANIMAL

C.R.: Différents contrôles ont été faits sur les données de poids. Ainsi, le module se charge de rechercher les données de poids qui pourraient avoir été enregistrées avec un ancien "num_animal" et si c'est le cas, il les met à jour en fonction du "num_animal" dont on dispose à ce moment (c'est-à-dire si le "num_animal" n'est pas sous la forme nr_lot + code_def).
Dans le cas où on connait un nouveau numéro ou code étude pour un animal, les tables CL_NUMERO et CL_CODE sont mises à jour.

Le module CONV_XYBION

D.: Les fichiers extractions de XYBION ou XYBANI.
Le fichier FICH_CONVERTI.
Le code étude concerné.

C.D.: - les fichiers extractions contiennent des données extraites de Xybion ou Xybani. Leur structure a été décrite en annexe (cfr annexe n° 7).
- le fichier FICH_CONVERTI est vide. Il a la structure décrite dans le module "ENTREES_XYBION".

R.: Le fichier FICH_CONVERTI contient toutes les données extraites du logiciel Xybion ou Xybani. Celles-çi concernent un ou plusieurs déterminants et donc un ou plusieurs paramètres biologiques pour chaque déterminant, cela pour une seule et même étude. Elles sont par ailleurs stockées de telle façon qu'elles soient utilisables par Powerhouse (compatibilité de format).

C.R.: les données contenues dans FICH_CONVERTI sont enregistrées séquentiellement suivant chaque extraction réalisée dans Xybion ou Xybani et concernant un déterminant à la fois. Ceci ne signifiant pas qu'elles soient triées par déterminant puisqu'il est possible de trouver des données concernant un même déterminant une fois au début du fichier FICH_CONVERTI et une fois vers la fin de celui-çi ! En effet on stocke successivement dans ce fichier les données d'une extraction pour un déterminant donné à la fois suivant la séquence suivie par l'utilisateur pour faire des extractions dans XYBION.

Le module TRANSFERT_TEST

D.: -Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES et FICH_CONVERTI.
-le code de l'étude considérée.

C.D.: - le fichier FICH_CONVERTI contient des données concernant une seule étude d'animaux en phase Test et/ou en phase Pretest. Sa structure a été décrite dans le module "ENTREES_XYBION".
- les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES sont vides. Leur structure a été décrite dans le module "ENTREES_XYBION".

R.: Les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES sont remplis avec les données transférées à partir du fichier FICH_CONVERTI.

C.R.: - le fichier ELEMCOMMUNS contient un seul enregistrement. Le seul élément qu'il contient à ce moment là est le code de l'étude.

- le fichier ELEMVARIABLES contient autant d'enregistrements qu'il existe d'animaux différents en phase Test seulement!
Les éléments qu'il contient à ce moment là, pour chacun des enregistrements, sont :
le numéro d'animal
le sexe

- le fichier ELEMHYPERVARIABLES contient autant d'enregistrements qu'il existe de résultats d'animaux en phase Test seulement, possédant un enregistrement dans le fichier ELEMVARIABLES. Des résultats non reliés à un animal de ELEMVARIABLES ne peuvent exister.
Les éléments qu'il contient à ce moment là, pour chacun des enregistrements, sont:

le déterminant
le paramètre biologique
le jour de l'étude
la valeur du résultat

Le module GARNISSAGE_QUAR

D.: -Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES, ANIMAL, PRETEST, TEST, NUM_MAX, CONDEXPERIMENT, RESULTATS, ETUDES.

C.D. - les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES sont remplis. Leur structure a été décrite dans le module "ENTREES_XYBION". Les données qu'ils contiennent sont syntaxiquement correctes. Elles sont complètes et cohérentes.

- le champs "etat" du fichier ELEMCOMMUNS a la valeur "Pretest" ou "Quar".

- les "num_animal" des données sont sous la forme correcte pour le garnissage.

- les fichiers ANIMAL, PRETEST, TEST, CONDEXPERIMENT, RESULTATS et ETUDES ont leur structure décrite en annexe (cfr.annexe n° 7).

- le fichier NUM_MAX contient le dernier numéro de fournisseur attribué séquentiellement.

R.: Les fichiers ETUDES, ANIMAL, PRETEST, TEST, NUM_MAX, CONDEXPERIMENT et RESULTATS sont mis à jour.

C.R.: - le fichier ETUDES est mis à jour à chaque garnissage. Il contient la date du début d'étude de chaque étude considérée. Il sert à de nombreux contrôles de cohérence.

- le fichier NUM_MAX a son unique champs (numéro) de son unique enregistrement qui est incrémenté de une unité si un nouveau fournisseur a été identifié dans les données qui ont servi au garnissage.

- les fichiers de la base de données ne se voient ajouter des enregistrements correspondant à ces données de garnissage que si et seulement si elles n' y sont pas déjà stockées !

Le module GARNISSAGE_RAT_TEST

D.: -Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES, ETUDES, ANIMAL, PRETEST, TEST, NUM_MAX, CONDEXPERIMENT, RESULTATS, TAB_CHANGEMENTS, CL_NUMERO.

C.D. - les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES sont remplis avec les données à transférer dans la base de données. Leur structure a été décrite dans le module ENTREES_XYBION. Les données qu'ils contiennent sont syntaxiquement correctes. Elles sont complètes et cohérentes.

- le champs "etat" du fichier ELEMCOMMUNS a la valeur "test".
  le champs "espece" du fichier ELEMCOMMUNS a la valeur "rat".

- les "num_animal" des données sont sous la forme correcte pour le garnissage.

- les fichiers ANIMAL, PRETEST, TEST, CONDEXPERIMENT, RESULTATS et ETUDES ont leur structure décrite en annexe (cfr.annexe n° 7).

- le fichier NUM_MAX contient le dernier numéro de fournisseur attribué séquentiellement.

- le fichier TAB_CHANGEMENTS contient la liste des numéros de lot, numéros définitifs, codes études bidons ou codes études définitifs (du moins ceux parmi eux dont on dispose à ce moment) pour chaque animal du fichier ELEMVARIABLES. Sa structure a été décrite dans le module "ENTREES_XYBION".

- le fichier CL_NUMERO a sa structure décrite dans le module "ENTREES_XYBION".

R.: Les fichiers ETUDES, ANIMAL, PRETEST, TEST, NUM_MAX, CONDEXPERIMENT, RESULTATS et CL_NUMERO sont mis à jour.

C.R.: - le fichier ETUDES est mis à jour à chaque garnissage. Il contient la date du début d'étude de chaque étude considérée. Il sert à de nombreux contrôles de cohérence.

- le fichier NUM_MAX a son unique champs (numéro) de son unique enregistrement qui est incrémenté de une unité si un nouveau fournisseur a été identifié dans les données qui ont servi au garnissage.

- les fichiers de la base de données ne se voient ajouter des enregistrements correspondant à ces données de garnissage que si et seulement si elles n' y sont pas déjà stockées !

- le fichier CL_NUMERO contient autant de nouveaux enregistrements qu'on connait de correspondance entre les numéros de lot et les numéros définitif pour les animaux dont les données ont été transférées dans la B.D.

Le module GARNISSAGE_NR_PRE_QUAR

D.: -Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES, ETUDES, ANIMAL, PRETEST, TEST, NUM_MAX, CONDEXPERIMENT, RESULTATS, TAB_CHANGEMENTS, CL_CODE.

C.D. - les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES sont remplis avec les données à transférer dans la base de données. Leur structure a été décrite dans le module "ENTREES_XYBION". Les données qu'ils contiennent sont syntaxiquement correctes. Elles sont complètes et cohérentes.

- le champs "etat" du fichier ELEMCOMMUNS a la valeur "Pretest". Le champs "espece" du fichier ELEMCOMMUNS a une valeur <> "rat".

- les "num_animal" des données sont sous la forme correcte pour le garnissage.

- les fichiers ANIMAL, PRETEST, TEST, CONDEXPERIMENT, RESULTATS et ETUDES ont leur structure décrite en annexe (cfr.annexe n° 7).

- le fichier NUM_MAX contient le dernier numéro de fournisseur attribué séquentiellement.

- le fichier TAB_CHANGEMENTS contient la liste des codes études bidons ou codes études définitifs (du moins ceux parmi eux dont on dispose à ce moment) pour chaque animal du fichier ELEMVARIABLES. Sa structure a été décrite dans le module "ENTREES_XYBION".

- le fichier CL_CODE a sa structure décrite dans le module "ENTREES_DONNEES_POIDS".

- des données de quarantaine concernant les mêmes animaux que ceux d'elemvariables sont déjà stockés dans la B.D.

R.: - Les fichiers ETUDES, ANIMAL, PRETEST, TEST, NUM_MAX, CONDEXPERIMENT, RESULTATS et CL_CODE sont mis à jour.

- les enregistrements du fichier RESULTATS ont leur clé (num_animal) qui est convertie avec une nouvelle valeur de clé si ils concernent les mêmes animaux que ceux impliqués dans le garnissage et si il existe des résultats les concernant stockés sous un num_animal de quarantaine (num_animal = nr_lot + code_etude_bidon).

C.R.: -le fichier ETUDES est mis à jour à chaque garnissage. Il contient la date du début d'étude de chaque étude considérée. Il sert à de nombreux contrôles de cohérence.

- le fichier NUM_MAX a son unique champs (numéro) de son unique enregistrement qui est incrémenté de une unité si un nouveau fournisseur a été identifié dans les données qui ont servi au garnissage.

- les fichiers de la base de données ne se voient ajouter des enregistrements correspondant à ces données de garnissage que si et seulement si elles n' y sont pas déjà stockées !

- le fichier CL_CODE contient autant de nouveaux enregistrements qu'on connait de correspondances entre les codes études bidons et les codes études définitifs pour les animaux dont les données ont été transférées dans la B.D. (ceci se fait donc uniquement dans le cas des non rongeurs).

Le module GARNISSAGE_NR_TEST_QUAR

D.: -Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES, ETUDES, ANIMAL, PRETEST, TEST, NUM_MAX, CONDEXPERIMENT, RESULTATS, TAB_CHANGEMENTS, CL_CODE et CL_NUMERO.

C.D. - les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES sont remplis avec les données à transférer dans la base de données. Leur structure a été décrite dans le module "ENTREES_XYBION". Les données qu'ils contiennent sont syntaxiquement correctes. Elles sont complètes et cohérentes.

- le champs "etat" du fichier ELEMCOMMUNS a la valeur "Test". Le champs "espece" du fichier ELEMCOMMUNS a une valeur <> "rat".

- les "num_animal" des données sont sous la forme

correcte pour le garnissage.

- les fichiers ANIMAL, PRETEST, TEST, CONDEXPERIMENT, RESULTATS et ETUDES ont leur structure décrite en annexe (cfr.annexe n° 7).

- le fichier NUM_MAX contient le dernier numéro de fournisseur attribué séquentiellement.

- le fichier TAB_CHANGEMENTS contient la liste des codes études bidons ou codes études définitifs (du moins ceux parmi eux dont on dispose à ce moment) pour chaque animal du fichier ELEMVARIABLES. Sa structure a été décrite dans le module "ENTREES_XYBION".

- le fichier CL_CODE a sa structure décrite dans le module "ENTREES_DONNEES_POIDS".

- le fichier CL_NUMERO a sa strusture décrite dans le module "ENTREES_XYBION".

- des données de quarantaine concernant les mêmes animaux que ceux du fichier ELEMVARIABLES, sont déjà stockées dans la B.D.

R.: - Les fichiers ETUDES, ANIMAL, PRETEST, TEST, NUM_MAX, CONDEXPERIMENT, RESULTATS, CL_CODE et CL_NUMERO sont mis à jour.

- les enregistrements du fichier RESULTATS ont leur clé (num_animal) qui est convertie avec une nouvelle valeur de clé si ils concernent les mêmes animaux que ceux impliqués dans le garnissage et si il existe des résultats les concernant stockés sous un num_animal de quarantaine (num_animal = nr_lot + code_etude_bidon).

C.R.: -le fichier ETUDES est mis à jour à chaque garnissage. Il contient la date du début d'étude de chaque étude considérée. Il sert à de nombreux contrôles de cohérence.

- le fichier NUM_MAX a son unique champs (numéro) de son unique enregistrement qui est incrémenté de une unité si un nouveau fournisseur a été identifié dans les données qui ont servi au garnissage.

- les fichiers de la base de données ne se voient ajouter des enregistrements correspondant à ces données de garnissage que si et seulement si elles n' y sont déjà pas stockées !

- le fichier CL_CODE contient autant de nouveaux enregistrements qu'on connait de correspondance entre les codes études bidons et les codes études définitifs pour les animaux dont les données ont été transférées dans la B.D. (ceci se fait donc uniquement dans le cas des non rongeurs).

- le fichier CL_NUMERO contient autant de nouveaux enregistrements qu'on connait de correspondance entre les numéros de lot et les numéros définitifs pour les animaux dont les données ont été transférées dans la B.D.

Le module TRANSFERT_PRETEST

D.: -Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES et FICH_CONVERTI.
-le code de l'étude considéré.

C.D.: - le fichier FICH_CONVERTI contient des données concernant une seule étude d'animaux en phase Test ou en phase Pretest. Sa structure a été décrite dans le module "ENTREES_XYBION".

- les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES sont vides. Leur structure a été décrite dans le module "ENTREES_XYBION".

R.: Les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES sont remplis avec les données transférées à partir du fichier FICH_CONVERTI.

C.R.: - le fichier ELEMCOMMUNS contient un seul enregistrement. Le seul élément qu'il contient à ce moment là est le code de l'étude.

- le fichier ELEMVARIABLES contient autant d'enregistrements qu'il existe d'animaux différents en phase Pretest seulement!
Les éléments qu'il contient à ce moment là sont :
le numéro d'animal
le sexe

- le fichier ELEMHYPERVARIABLES contient autant d'enregistrements qu'il existe de résultats d'animaux en phase Pretest seulement, possédant un enregistrement dans le fichier ELEMVARIABLES. Des résultats non reliés à un animal de ELEMVARIABLES ne peuvent exister.
Les éléments qu'il contient à ce moment là sont:

le déterminant
le paramètre biologique
le jour de l'étude
la valeur du résultat

Le module ELEMENTS_COMMUNS_ION

D.: Les fichiers ELEMCOMMUNS, ANIMAL, ETUDES, RACE, ESPECE, TYPE_ALIMENT, VOIE_INJECTION, TYPE_ECHANTILLON, VEHICULE_UTILISE, TABLE_MESURE.

C.D.: - le fichier ELEMCOMMUNS est pré-rempli avec les données extraites du logiciel Xybion pour une étude donnée. Il contient le code de l'étude.

- les fichiers RACE, ESPECE, TYPE_ALIMENT, VOIE_INJECTION, TYPE_ECHANTILLON, VEHICULE_UTILISE sont constitués d'un ensemble d'enregistrements (à un seul champs) reprenant pour chaque type d'élément, l'ensemble des valeurs permises pour celui-ci. Cet ensemble peut être modifié par l'utilisateur en utilisant le module "MENU_MODIF_TABLES".

- la structure des fichiers ANIMAL et ETUDES est décrite en annexe (cfr.annexe n° 7).

- la structure du fichier TABLE_MESURE a été décrite dans le module "ENTREES_DONNEES_POIDS".

R.: - les champs des enregistrements du fichier ELEMCOMMUNS sont remplis ou mis à jour.

- les fichiers ELEMVARIABLES et ELEMHYPERVARIABLES sont remplis par appel au module "ELEMENTS_VARIABLES_ION" et "ELEMENTS_HYPERVARIABLES_ION". Les numéros d'animaux de ces fichiers (num_animal) sont mis à jour.

C.R.: - les champs num_animal de ELEMVARIABLES et ELEMHYPERVARIABLES sont mis à jour en fonction du champs "etat" d'ELEMCOMMUNS:
soit par l'appel au module "MAJ_NUM_ANIMAL_TEST"
s'il s'agit de données de la phase Test
soit par l'appel au module "MAJ_NUM_ANIMAL_PRETEST"
s'il s'agit de données de la phase Pretest.

Ils valent finalement num_animal = nr_lot + code_def.
Le contenu des champs des trois fichiers est syntaxiquement correct.

toute une série de contrôles est effectuée :
* si l'étude a déjà été traitée auparavant (données présentes dans la B.D.), les éléments suivants sont automatiquement pré-garnis :
date_début_étude, race et espece.
* pour chaque champ, les types de valeur permise sont répertoriés dans une table spécifique. De plus, des valeurs par défaut sont disponibles à tout instant.

APPELS: ce module appelle les modules
ELEMENTS_VARIABLES_ION,
MAJ_NUM_ANIMAL_PRETEST et
MAJ_NUM_ANIMAL_TEST.

Le module ELEMENTS_VARIABLES_ION

D.: Les fichiers ELEMCOMMUNS,ELEMVARIABLES, TAB_CHANGEMENTS, FOURNISSEUR, NUM_MAX, TABLE_MESURE, ELEMHYPERVARIABLES, PRETEST, CL_NUMERO, ANIMAL.

C.D.: - le fichier ELEMVARIABLES est pré-rempli avec les données extraites du logiciel Xybion et concernant une étude déterminée. Il contient ainsi le numéro d'animal et le sexe de chaque animal.

- le fichier ELEMCOMMUNS contient les données communes aux résultats des animaux extraits de Xybion.

- les autres fichiers ont été décrits ci-dessus.

R.: - les champs des enregistrements du fichier ELEMVARIABLES sont remplis ou mis à jour.

- les champs des enregistrements du fichier ELEMHYPERVARIABLES sont remplis ou mis à jour par appel au module "ELEMENTS_HYPERVARIABLES_ION".

- le fichier TAB_CHANGEMENTS est mis à jour avec les différents numéros et codes études pour les animaux d'ELEMVARIABLES.

- le fichier FOURNISSEUR est mis à jour

C.R.: les contrôles effectués à ce niveau consistent en :
- recherche de données éventuellement déjà présentes dans la B.D. pour les afficher en essayant les différentes clés possibles pour un animal (différents "num_animal").

- contrôle des numéros de lot et des numéros définitifs par la table CL_NUMERO

- contrôle que la date d'entrée en quarantaine est bien inférieure à la date du début de l'étude car on est en phase Test ou Pretest.

- contrôle que la date d'entrée n'est pas supérieure à la date du résultat de ELEMHYPERVARIABLES.

- contrôle que le fournisseur existe ou bien met à jour la table.

- pour chaque champs, il y a un contrôle des valeurs permises si c'est possible en accédant aux tables spécifiques.

- en ce qui concerne les unités de mesure, toutes les données introduites sont transformées de telle façon qu'on a toujours (où que ce soit dans la B.D.) :
  l'âge exprimé en nombre de jours.
  le volume administré exprimé en nombre de millilitres.
  la période d'acclimatation en nombre de jours.
  la durée de quarantaine en nombre de jours.

Le module ELEMENTS_HYPERVARIABLES_ION

D.: Les fichiers ELEMVARIABLES, TAB_CHANGEMENTS, ELEMHYPERVARIABLES, PARAMBIOL.

C.D.: le fichier ELEMHYPERVARIABLES est rempli avec les données extraites du logiciel Xybion pour une étude donnée (nom_determinant, nom_param_bio, jour_etude, valeur_res, date_res).

R.: - les champs des enregistrements du fichier ELEMHYPERVARIABLES sont remplis ou mis à jour.

- la table TAB_CHANGEMENTS est remplie avec les différents numéros et codes d'étude pour les éléments d'ELEMVARIABLES.

C.R.: les contrôles effectués sont:
- contrôle pour voir si le paramètre biologique et la méthode d'analyse existent dans la table des valeurs permises.

- contrôle pour voir si les valeurs introduites ne l'ont pas déjà été pour ce même animal et cette même étude.

en ce qui concerne les unités de mesures, on a toujours

le jour_etude exprimé en nombre de jours
la valeur du résultat exprimé en fonction de
l'unité de mesure
correspondant au paramètre
biologique considéré.

Le module MAJ_NUM_ANIMAL_TEST

D.: les fichiers ELEMVARIABLES, TAB_CHANGEMENTS,
ELEMHYPERVARIABLES,

C.D.: - Le fichier TAB_CHANGEMENTS contient la correspondance
entre les numéros (de lot et définitif) ainsi que des
codes études (code_bidon et code_def) pour une étude.

- les champs "num_animal" de tous les enregistrements
de ELEMVARIABLES et ELMEMHYPERVARIABLES sont sous la
forme nr_def + code_def.

R.: les fichiers ELEMVARIABLES et ELMEMHYPERVARIABLES ont
pour chacun de leur enregistrement, le champs num_aniaml
qui est mis à jour.

C.R.: les champs "num_animal" sont mis à jour de telle façon
qu'on ait num_animal sous la forme nr_lot + code_def.

Le module MAJ_NUM_ANIMAL_PRETEST

D.: les fichiers ELEMVARIABLES, ELEMCOMMUNS,
ELEMHYPERVARIABLES, TAB_CHANGEMENTS.

C.D.: - Le fichier TAB_CHANGEMENTS contient la correspondance
entre les numéros (de lot et définitif) ainsi que des
codes études (code_bidon et code_def) pour une étude.

- les champs "num_animal" de tous les enregistrements
de ELEMVARIABLES et ELMEMHYPERVARIABLES sont sous la
forme nr_lot + code_def.

R.: les fichiers ELEMVARIABLES et ELMEMHYPERVARIABLES ont
pour chacun de leur enregistrement, le champs num_aniaml
qui est mis à jour.

C.R.: les champs "num_animal" sont mis à jour de telle façon qu'on ait num_animal sous la forme nr_lot + code_def.

Le module TRANSFERT_ANI

D.: -Les fichiers ELEMCOMMUNS, ELEMVARIABLES, ELEMHYPERVARIABLES et FICH_CONVERTI.
-le code de l'étude considéré.

C.D.: - le fichier FICH_CONVERTI contient des données concernant une seule étude d'animaux en phase quarantaine .Sa structure a été décrite dans le module "ENTREES_XYBION".

- les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES sont vides. Leur structure a été décrite dans le module "ENTREES_XYBION".

R.: Les fichiers ELEMCOMMUNS, ELEMVARIABLES et ELEMHYPERVARIABLES sont remplis avec les données transférées à partir du fichier FICH_CONVERTI.

C.R.: - le fichier ELEMCOMMUNS contient un seul enregistrement. Le seul élément qu'il contient à ce moment là est le code de l'étude.

- le fichier ELEMVARIABLES contient autant d'enregistrements qu'il existe d'animaux différents en phase Quarantaine seulement!
Les éléments qu'il contient à ce moment là sont :
le numéro d'animal
le sexe

- le fichier ELEMHYPERVARIABLES contient autant d'enregistrements qu'il existe de résultats d'animaux en phase Quarantaine seulement, possédant un enregistrement dans le fichier ELEMVARIABLES. Des résultats non reliés à un animal de ELEMVARIABLES ne peuvent exister.
Les éléments qu'il contient à ce moment là sont:
le déterminant
le paramètre biologique
le jour de l'étude
la valeur du résultat

Le module ELEMENTS_COMMUNS_ANI

D.: Les fichiers ELEMCOMMUNS, ANIMAL, ETUDES, RACE, ESPECE, TYPE_ALIMENT, VOIE_INJECTION, TYPE_ECHANTILLON, VEHICULE_UTILISE, TABLE_MESURE.

C.D.: - le fichier ELEMCOMMUNS est pré-rempli avec les données extraites du logiciel Xybani pour une étude donnée. Il contient le code de l'étude.

- les fichiers RACE, ESPECE, TYPE_ALIMENT, VOIE_INJECTION, TYPE_ECHANTILLON, VEHICULE_UTILISE sont constitués d'un ensemble d'enregistrements (à un seul champs) reprenant pour chaque type d'élément, l'ensemble des valeurs permises pour celui-ci. Cet ensemble peut être modifiés par l'utilisateur en utilisant le module "MENU_MODIF_TABLES".

- la structure des fichiers ANIMAL et ETUDES est décrite en annexe (cfr.annexe n° 7).

- la structure du fichier TABLE_MESURE a été décrite dans le module "ENTREES_DONNEES_POIDS".

R.: - les champs des enregistrements du fichier ELEMCOMMUNS sont remplis ou mis à jour.

- les fichiers ELEMVARIABLES et ELEMHYPERVARIABLES sont remplis par appel au module ELEMENTS_VARIABLES_ANI et ELEMENTS_HYPERVARIABLES_ANI.

C.R.: - les champs num_animal de ELEMVARIABLES et ELEMHYPERVARIABLES sont sous la forme num_animal = nr_lot + code_etude_bidon. Le contenu des champs des trois fichiers est syntaxiquement correct.

- toute une série de contrôles est effectuée :
  * si l'étude a déjà été traitée auparavant (données présentes dans la B.D.), les éléments suivants sont automatiquement pré-garnis :
    date_début_étude, race et espece.
  * pour chaque champ, les types de valeur permise sont répertoriés dans une table spécifique. De plus, des valeurs par défaut sont disponibles à tout instant.

APPELS: ce module appelle le module ELEMENTS_VARIABLES_ANI

Le module ELEMENTS_VARIABLES_ANI

D.: Les fichiers ELEMCOMMUNS,ELEMVARIABLES, TAB_CHANGEMENTS, FOURNISSEUR, NUM_MAX, TABLE_MESURE, ELEMHYPERVARIABLES, PRETEST, CL_NUMERO, ANIMAL.

C.D.: - le fichier ELEMVARIABLES est pré-rempli avec les données extraites du logiciel Xybani et concernant une étude déterminée. Il contient ainsi le numéro d'animal et le sexe de chaque animal.

- le fichier ELEMCOMMUNS contient les données communes aux résultats des animaux extraits de Xybion.

- les autres fichiers ont été décrits ci-dessus.

R.: - les champs des enregistrements du fichier ELEMVARIABLES sont remplis ou mis à jour.

- les champs des enregistrements du fichier ELEMHYPERVARIABLES sont remplis ou mis à jour par appel au module ELEMENTS_HYPERVARIABLES_ANI.

- le fichier TAB_CHANGEMENTS est mis à jour avec les différents numéros et codes études pour les animaux d'ELEMVARIABLES.

- le fichier FOURNISSEUR est mis à jour.

C.R.: les contrôles effectués à ce niveau consistent en :
- recherche de données éventuellement déjà présentes dans la B.D. pour les afficher en essayant les différentes clés possibles pour un animal (différents "num_animal").

- contrôle que la date d'entrée en quarantaine est bien inférieure à la date du début de l'étude bidon.

- contrôle que la date d'entrée n'est pas supérieure à la date du résultat de ELEMHYPERVARIABLES.

- contrôle que le fournisseur existe ou bien met à jour la table.

- pour chaque champs, il y a un contrôle des valeurs permises si c'est possible en accédant aux tables spécifiques.

- en ce qui concerne les unités de mesure, toutes les données introduites sont transformées de telle façon qu'on a toujours (où que ce soit dans la B.D.) :
l'âge exprimé en nombre de jours.
le volume administre exprimé en nombre de

millilitres.
la période d'acclimatation en nombre de jours.
la durée de quarantaine en nombre de jours.

Le module ELEMENTS_HYPERVARIABLES_ANI

D.: Les fichiers ELEMVARIABLES, TAB_CHANGEMENTS, ELEMHYPERVARIABLES, PARAMBIOL.

C.D.: le fichier ELEMHYPERVARIABLES est rempli avec les données extraites du logiciel Xybani pour une étude donnée (nom_determinant, nom_param_bio, jour_etude, valeur_res, date_res).

R.: - les champs des enregistrements du fichier ELEMHYPERVARIABLES sont remplis ou mis à jour.

- la table TAB_CHANGEMENTS est remplie avec les différents numéros de lot et codes d'étude bidons pour les éléments d'ELEMVARIABLES.

C.R.: les contrôles effectués sont:
- contrôle pour voir si le paramètre biologique et la méthode d'analyse existent dans la table des valeurs permises.

- contrôle pour voir si les valeurs introduites ne l'ont pas déjà été pour ce même animal et cette même étude.

en ce qui concerne les unités de mesures, on a toujours

le jour_etude exprimé en nombre de jours
la valeur du résultat exprimé en fonction de l'unité de mesure correspondant au paramètre biologique considéré.

Le module ELEMENTS_COMMUNS_MAN

D.: Les fichiers ELEMCOMMUNS, ANIMAL, ETUDES, RACE, ESPECE, TYPE_ALIMENT, VOIE_INJECTION, TYPE_ECHANTILLON, VEHICULE_UTILISE, TABLE_MESURE.

C.D.: - le fichier ELEMCOMMUNS est vide ou bien contient des données concernant des résultats pour une étude donnée.

- les fichiers RACE, ESPECE, TYPE_ALIMENT, VOIE_INJECTION, TYPE_ECHANTILLON, VEHICULE_UTILISE sont constitués d'un ensemble d'enregistrements (à un seul champs) reprenant pour chaque type d'éléments, l'ensemble des valeurs permises pour celui-ci. Cest ensemble peut être modifiés par l'utilisateur en utilisant le module "MENU_MODIF_TABLES".

- la structure des fichiers ANIMAL et ETUDES est décrite en annexe (cfr.annexe n° 7).

- la structure du fichier TABLE_MESURE a été décrite dans le module "ENTREES_DONNEES_POIDS".

R.: - les champs des enregistrements du fichier ELEMCOMMUNS sont remplis ou mis à jour.

- les fichiers ELEMVARIABLES et ELEMHYPERVARIABLES sont remplis par appel au module "ELEMENTS_VARIABLES_MAN" et "ELEMENTS_HYPERVARIABLES_MAN". Les numéros d'animaux de ces fichiers (num_animal) sont mis à jour.

C.R.: - les champs num_animal de ELEMVARIABLES et ELEMHYPERVARIABLES sont mis à jour en fonction du champs "etat" d'ELEMCOMMUNS:
soit par l'appel au module "MAJ_NUM_ANIMAL_TEST"
s'il s'agit de données de la phase Test
soit par l'appel au module " MAJ_NUM_ANIMAL_PRETEST"
s'il s'agit de données de la phase Pretest.

Ils valent finalement num_animal = nr_lot + code_def si on est en phase Test ou Prestest ou bien num_animal = nr_lot + code_etude_bidon si on est en phase Quarantaine. Le contenu des champs des trois fichiers est syntaxiquement correct.

- toute une série de contrôles est effectuée :
* si l'étude a déjà été traitée auparavant (données présentes dans la B.D.), les éléments suivants sont automatiquement pré-garnis :
date_début_étude, race et espece.
* pour chaque champ, les types de valeurs permises sont répertoriés dans une table spécifique.

De plus, des valeurs par défaut sont disponibles à tout instant.

APPELS: ce module appelle les modules ELEMENTS_VARIABLES_MAN, MAJ_NUM_ANIMAL_PRETEST et MAJ_NUM_ANIMAL_TEST.

Le module ELEMENTS_VARIABLES_MAN

D.: Les fichiers ELEMCOMMUNS,ELEMVARIABLES, TAB_CHANGEMENTS, FOURNISSEUR, NUM_MAX, TABLE_MESURE, ELEMHYPERVARIABLES, PRETEST, CL_NUMERO, ANIMAL.

C.D.: - le fichier ELEMVARIABLES est vide ou bien rempli avec des données concernant une étude déterminée.

- le fichier ELEMCOMMUNS contient les données communes aux résultats des animaux d'une étude.

- les autres fichiers ont été décrits ci-dessus.

R.: - les champs des enregistrements du fichier ELEMVARIABLES sont remplis ou mis à jour.

- les champs des enregistrements du fichier ELEMHYPERVARIABLES sont remplis ou mis à jour par appel au module "ELEMENTS_HYPERVARIABLES_MAN".

- le fichier TAB_CHANGEMENTS est mis à jour avec les différents numéros et codes études pour les animaux d'ELEMVARIABLES.

- le fichier FOURNISSEUR est mis à jour.

C.R.: les contrôles effectués à ce niveau consistent en :
- recherche de données éventuellement déjà présentes dans la B.D. pour les afficher en essayant les différentes clés possibles pour un animal (différents "num_animal").

- contrôle des numéros de lot et des numéros définitifs par la table CL_NUMERO

- contrôle que la date d'entrée en quarantaine est bien inférieure à la date du début de l'étude si on est en phase Test ou Pretest.

- contrôle que la date d'entrée n'est pas supérieure à la date du résultat de ELEMHYPERVARIABLES.

- contrôle que le fournisseur existe ou bien met à jour la table.

- pour chaque champs, il y a un contrôle des valeurs permises si c'est possible en accédant aux tables spécifiques.

- en ce qui concerne les unités de mesure, toutes les données introduites sont transformées de telle façon qu'on a toujours (où que ce soit dans la B.D.) :
  l'âge exprimé en nombre de jours.
  le volume administré exprimé en nombre de millilitres.
  la période d'acclimatation en nombre de jours.
  la durée de quarantaine en nombre de jours.

Le module ELEMENTS_HYPERVARIABLES_MAN

D.: Les fichiers ELEMVARIABLES, TAB_CHANGEMENTS, ELEMHYPERVARIABLES, PARAMBIOL.

C.D.: le fichier ELEMHYPERVARIABLES est vide ou bien rempli avec des données concernant une étude donnée.

R.: - les champs des enregistrements du fichier ELEMHYPERVARIABLES sont remplis ou mis à jour.

- la table TAB_CHANGEMENT est remplie avec les différents numéros et codes d'étude pour les éléments d'ELEMVARIABLES.

C.R.: les contrôles effectués sont:
- contrôle pour voir si le paramètre biologique et la méthode d'analyse existent dans la table des valeurs permises.

- contrôle pour voir si les valeurs introduites ne l'ont pas déjà été pour ce même animal et cette même étude.

en ce qui concerne les unités de mesures, on a toujours

le jour_etude exprimé en nombre de jours
la valeur du résultat exprimé en fonction de l'unité de mesure correspondant au paramètre biologique considéré.

Le module AJOUT_GENERAL

D.: Les fichiers DETERMINANT et PARAMBIOL.

C.D.: - les fichiers DETERMINANT et PARAMBIOL ont leur structure décrite en annexe (cfr.annexe n° 7).

R.: - le fichier DETERMINANT est mis à jour.
- le fichier PARAMBIOL est mis à jour.

C.R.: - le fichier DETERMINANT contient le déterminant qu'on vient d'ajouter ou contient un enregistrement en moins, correspondant au déterminant qu'on vient d'effacer.

- le fichier PARAMBIOL contient le paramètre biologique qu'on vient d'ajouter ou contient un enregistrement en moins, correspondant au paramètre biologique qu'on vient d'effacer.

APPEL: ce module appelle les modules AJOUT_DETERMINANT et AJOUT_PARAMETRE.

Le module AJOUT_DETERMINANT

D.: Les fichiers DETERMINANT et PARAMBIOL.

C.D.: - les fichiers DETERMINANT et PARAMBIOL ont leur structure décrite en annexe (cfr.annexe n° 7).

R.: - le fichier DETERMINANT est mis à jour.
- le fichier PARAMBIOL est mis à jour.

C.R.: - le fichier DETERMINANT contient le déterminant qu'on vient d'ajouter ou contient un enregistrement en moins, correspondant au déterminant qu'on vient d'effacer.

- le fichier PARAMBIOL contient le paramètre biologique se rapportant au déterminant qu'on vient d'ajouter.

Le module AJOUT_PARAMETRE

D.: Les fichiers DETERMINANT et PARAMBIOL.

C.D.: - les fichiers DETERMINANT et PARAMBIOL ont leur structure décrite en annexe (cfr.annexe n° 7).

R.: -le fichier PARAMBIOL est mis à jour.

C.R.: - le fichier PARAMBIOL contient le paramètre biologique qu'on vient d'ajouter ou contient un enregistrement en moins, correspondant au paramètre biologique qu'on vient d'effacer. Ce paramètre biologique ne peut être ajouté que s'il est rattaché à un déterminant.

Le module MENU_MODIF_TABLES

D.: les fichiers RACE, ESPECE, TYPE_ALIMENT, VOIE_INJECTION, TYPE_ECHANTILLON, VEHICULE_UTILISE.

C.D.: la structure de ces fichiers est donnée en annexe (cfr.annexe n° 7).

R.: un ou plusieurs enregistrements nouveaux sont créés pour un ou plusieurs de ces fichiers.

C.R.: les tables de valeurs permises correspondant à ces fichiers sont donc actualisées pour les contrôles futurs.

APPELS : ce module appelle les modules RACE, ESPECE, TYPE_ALIMENT, VOIE_INJECTION, TYPE_ECHANTILLON, VEHICULE_UTILISE

Le module VEHICULE_UTILISE

D.: le fichier VEHICULE_UTILISE
C.D.: sa structure est décrite en annexe (cfr.annexe n° 7).
R.: le fichier VEHICULE_UTILISE est modifié.
C.R.: la table des valeurs permises pour les types de véhicules utilisés est mise à jour.

Le module TYPE_ECHANTILLON
---
cfr.description ci-dessus

Le module VOIE_INJECTION
---
cfr.description ci-dessus

Le module TYPE_ALIMENT
---
cfr.description ci-dessus

Le module ESPECE
---

cfr.description ci-dessus

Le module RACE
---

cfr.description ci-dessus

Le module MENU_MAJ_BD

D.: l'ensemble des fichiers de la base de données
C.D.: - le contenu de ces fichiers est cohérent.
- leur structure est décrite en annexe (cfr.annexe nº 7).
R.: un ou plusieurs des fichiers de la B.D. sont mis à jour.
C.R. les fichiers mis à jour sont cohérents

PARTIE B : Exploitation de la B.D.

1. Module "Organisation de la requête"

Il s'agit d'un programme en PASCAL standard placé dans la librairie de modules de POWERHOUSE.

Données : - les dix critères de biologie clinique qui ont été introduits par l'utilisateur (voir point 4.3.2 par. 1)

- les variables qtp_1_def,qtp_2_def,qtp_3_def, qtp_4_def,qtp_2_1_def,qtp_3_1_def,qtp_3_2_def, qtp_4_1_def,qtp_4_2_def,qtp_4_3_def,qtp_5_4_def, qtp_6_4_def,qtp_6_5_def,qtp_5_4_fin_def, qtp_4_1_fin_def,qtp_4_2_fin_def,qtp_4_3_fin_def, qtp_4_fin_def.

Conditions sur les données : toutes les variables ci-dessus valent "F" (pour FALSE car les booléens n'existe pas).

Résultats : les variables décrites ci-dessus correspondant à la requête de recherche ont été modifiées (mises à "T" pour TRUE).

Conditions sur les résultats :

- qtp_1_def = "T" si le critère "Date-entrée" a été spécifié.
- qtp_2_def = "T" si le critère "Date-entrée" n'a pas été spécifié et soit un ou plusieurs critère(s) parmi "Espèce","Race","Sexe" a(ont) été spécifié(s).
- qtp_3_def = "T" si le critère "Fournisseur" a été spécifié et aucun des critères "Date-entrée","Espèce", "Race","Sexe" n'a été spécifié.
- qtp_4_def = "T" si au moins un des critères "Méthode-analyse","Type-échantillon","Etat-alimentaire","Age" a été spécifié et aucun des critères "Date-entrée","Espèce" ,"Race","Sexe" et "Fournisseur" ne l'a été . Il faut rappeler ici que le critère "Nom-paramètre-bio" doit toujours être fourni par l'utilisareur.
- qtp_2_1_def = "T" si un ou plusieurs critère(s) parmi "Espèce","Race","Sexe" a(ont) été spécifié(s) et si le critère "date-entrée" l'a été.
- qtp_3_1_def = "T" si le critère "Fournisseur" a été spécifié ainsi que le critère "Date-entrée" alors qu'aucun des critères "Espèce","Race","Sexe" ne l'a été.
- qtp_3_2_def = "T" si le critère "Fournisseur" a été spécifié ainsi qu'au moins un des critères "Espèce", "Race","Sexe".

- qtp_4_1_def = "T" si le critère "Date-entrée" a été spécifié et au moins un des critères "Méthode-analyse", "Type-échantillon","Etat-alimentaire" alors qu'aucun des critères "Espèce","Race","Sexe","Fournisseur" ne l'a été.
- qtp_4_2_def = "T" si un ou plusieurs critère(s) parmi "Espèce","Race","Sexe" a(ont) été spécifié(s)et au moins un parmi "Méthode-analyse","Type-échantillon","Etat-alimentaire" et "Age" alors que "Fournisseur" non.
- qtp_4_3_def = "T" si le critère "Fournisseur" a été spécifié et au moins un des critères "Méthode-analyse", "Type-échantillon","Etat-alimentaire".
- qtp_5_4_def = "T" si au moins un des trois critères "Méthode-analyse","Type-échantillon" et "Etat-alimentaire" a été spécifié et que le critère "Age" a été spécifié.
- qtp_6_4_def = "T" si le critère "Age" a été spécifié et qu'aucun des critères "Méthode-analyse","Type-échantillon","Etat-alimentaire" ne l'a été.
- qtp_6_5_def = "T" si le critère "Age" a été spécifié et au moins un parmi "Méthode-analyse","Type-échantillon", "Etat-alimentaire".
- qtp_5_4_fin_def = "T" si au moins un des critères "Méthode-analyse","Type-échantillon","Etat-alimentaire" a été spécifié et si le critère "Age" ne l'a pas été (justifie le mot fin dans la variable).
- qtp_4_1_fin_def = "T" si seulement le critère "Date-entrée" a été spécifié (et le paramètre !).
- qtp_4_2_fin_def = "T" si un ou plusieurs critères parmi "Espèce","Race","Sexe" a(ont) été spécifié(s) et si aucun des critères "Méthode-analyse","Type-échantillon","Etat-alimentaire","Age" et "Fournisseur" n' a été spécifié.
- qtp_4_3_fin_def = "T" si le critère "Fournisseur" a été spécifié et si aucun des critères "Méthode-analyse", "Type-échantillon","Etat-alimentaire" et "Age" ne l'a été.
- qtp_4_fin_def = "T" si aucun critère n'a été spécifié (hormis le "Nom-paramètre-bio" qui est obligatoire !).

2. Module "Recherche"

Comme il a été dit dans la conception logique , il s'agit d'un ensemble de sous-modules.

Les sous-modules de a) à u) sont implémentés sous la forme de QTP. Les sous-modules restants ont été implémentés sous la forme de QUIZ. La figure 32 représente les modules a) à u) et les fichiers sur lesquels ils travaillent et les fichiers obtenus après leur éxécution.

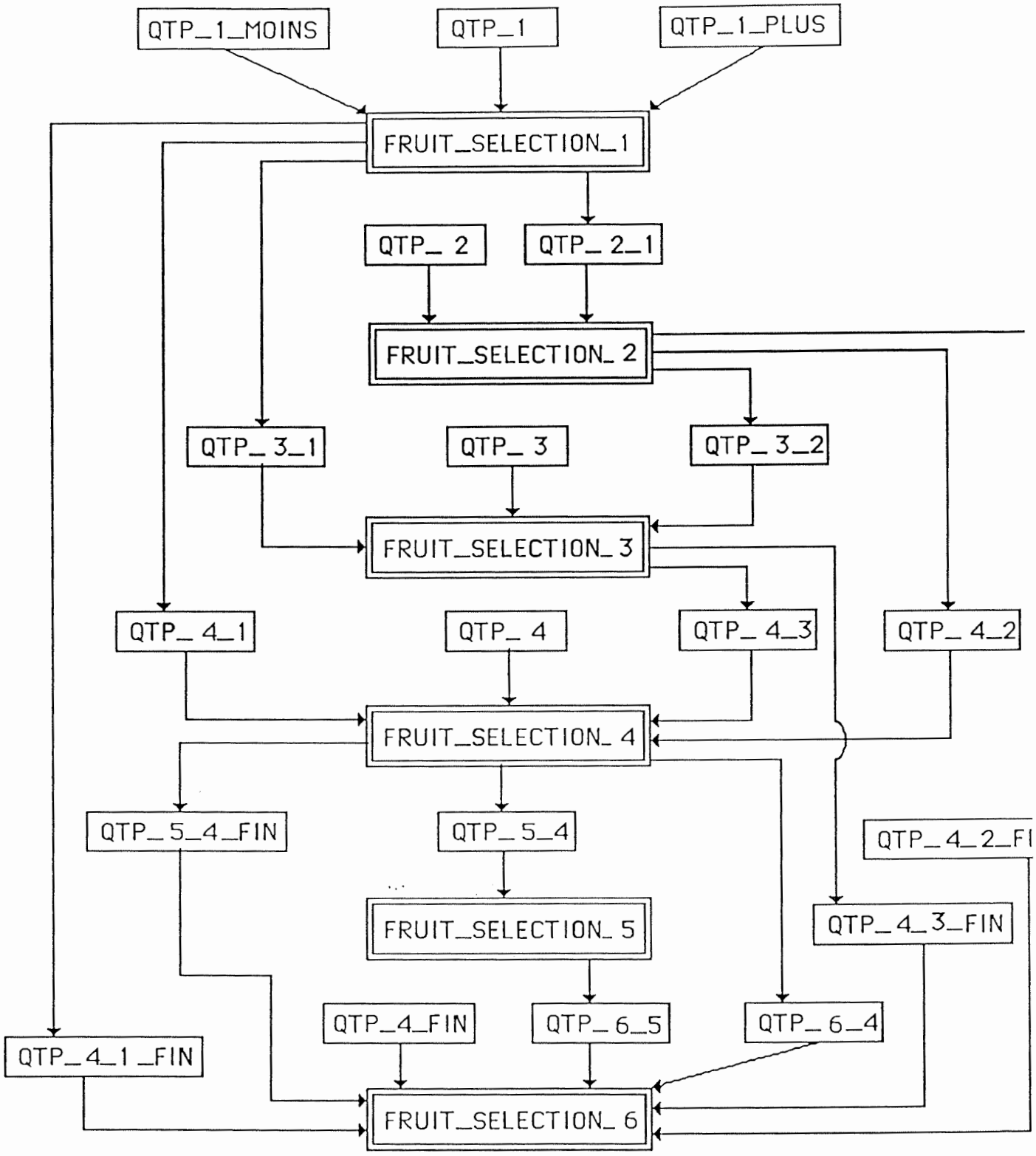


FIGURE 32: Enchaînement des modules de recherche pour les critères de bio. clini

Fichiers

Modules

Le paramètre biologique est toujours spécifié (le champ "Nom_param_bio" de REQUETE_BIOL est toujours garni;voir ci-après) et ne sera donc pas répété à chaque fois.

a) Sous-module QTP_1

Il donne la liste des animaux entrés à une date déterminée dans l'animalerie.

D. : le fichier REQUETE_BIOL

C. D. : - le fichier REQUETE_BIOL a la structure suivante :
+ Espèce
+ Race
+ Nom_fournisseur
+ Sexe
+ Age_animal
+ Date_entree
+ Etat_alimentaire
+ Type_echantillon
+ Nom_param_bio
+ Methode_analyse
+ Operateur_date
+ Operateur_age
+ Typestat
+ Unite_mesure

- operateur_age vaut " = "

- date_entrée est spécifié

R. : fichier FRUIT_SELECTION_1

C. R. : ce fichier contient pour les animaux sélectionnés :
+ le numéro d'animal (num_animal)
+ le numéro de fournissuer (num_fournisseur)
+ l'espèce (espece)
+ la race (race)
+ le sexe (sexe)

b) Sous-module QTP_1_MOINS

Il donne la liste des animaux entrés avant une date déterminée dans l'animalerie.

D. : le fichier REQUETE_BIOL

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a)

- operateur_age vaut " < "

- date_entrée est spécifié

R. : fichier FRUIT_SELECTION_1

C. R. : sa strucure est donnée au point a)

c) Sous-module QTP_1_PLUS

Il donne la liste des animaux entrés après une date déterminée dans l'animalerie.

D. : le fichier REQUETE_BIOL

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a)

- operateur_age vaut " > "

- date_entrée est spécifié

R. : fichier FRUIT_SELECTION_1

C. R. : sa strucure est donnée au point a)

d) Sous-module QTP_2

Il donne la liste des animaux pour lesquels aucune date d'entrée dans l'animalerie n'a été demandée et pour lesquels soit l'espèce , la race , le sexe ou une combinaison des trois a été demandé .

D. : le fichier REQUETE_BIOL

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a)

- au moins un des champs "Espece","Race","Sexe" de REQUETE_BIOL est spécifié

- le champ "Date_entrée" est non spécifié

R. : fichier FRUIT_SELECTION_2

C. R. : ce fichier contient pour les animaux sélectionnés :
+ le numéro d'animal (num_animal)
+ le numéro de fournisseur (num_fournisseur)

e) Sous-module QTP_2_1

Il donne la liste des animaux pour lesquels une date d'entrée dans l'animalerie a été demandée (sélection selon ce critère effectuée par un autre sous-module) et pour

lesquels soit l'espèce , la race , le sexe ou une combinaison des trois a été demandé .

D. : - le fichier REQUETE_BIOL
     - le fichier FRUIT_SELECTION_1

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) ainsi que FRUIT_SELECTION_1

- au moins un des champs "Espece","Race","Sexe" de REQUETE_BIOL est spécifié

- le champ "Date_entrée" est spécifié et FRUIT_SELECTION_1 contient le résultat le la recherche selon ce critère

R. : fichier FRUIT_SELECTION_2

C. R. : ce fichier contient pour les animaux sélectionnés :
+ le numéro d'animal (num_animal)
+ le numéro de fournisseur (num_fournisseur)

f) Sous-module QTP_3

Il donne la liste des animaux pour lesquels un fournisseur a été demandé et pour lesquels ni l'espèce , la race , le sexe , une date d'entrée dans l'animalerie n'a été demandé .

D. : le fichier REQUETE_BIOL

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a)

- le champ "Fournisseur" de REQUETE_BIOL est spécifié

- aucun des champs "Date-entrée","Espece","Race", "Sexe" de REQUETE_BIOL n'est spécifié

R. : fichier FRUIT_SELECTION_3

C. R. : ce fichier contient :
+ le numéro d'animal (num_animal)

g) Sous-module QTP_3_1

Il donne la liste des animaux pour lesquels un fournisseur et une date d'entrée dans l'animalerie (sélection selon ce dernier critère réalisée dans un autre sous-module) ont été demandés et pour lesquels ni l'espèce , la race , le sexe , n'a été demandé .

D. : - le fichier REQUETE_BIOL
- le fichier FRUIT_SELECTION_1

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) ainsi que FRUIT_SELECTION_1

- le champ "Fournisseur" de REQUETE_BIOL est spécifié

- le champ "Date-entrée de REQUETE_BIOL est spécifié et FRUIT_SELECTION_1 contient le résultat de la recherche selon ce critère

- aucun des champs "Espece","Race", "Sexe" de REQUETE_BIOL n'est spécifié

R. : fichier FRUIT_SELECTION_3

C. R. : ce fichier contient :
+ le numéro d'animal (num_animal)

h) Sous-module QTP_3_2

Il donne la liste des animaux pour lesquels un fournisseur a été demandé et au moins un ou plusieurs des critères parmi l'espèce , la race , le sexe , a(ont) été demandé(s) (sélection selon ce dernier ou ces derniers critères réalisées dans un autre sous-module).

D. : - le fichier REQUETE_BIOL
- le fichier FRUIT_SELECTION_2

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) alors que FRUIT_SELECTION_2 est décrit au point d) .

- le champ "Fournisseur" de REQUETE_BIOL est spécifié

- un au moins des champs "Espece","Race", "Sexe" de REQUETE_BIOL a été spécifié et FRUIT_SELECTION_2 contient le résultat de la recherche selon ces critères

R. : fichier FRUIT_SELECTION_3

C. R. : ce fichier contient :
+ le numéro d'animal (num_animal)

i) Sous-module QTP_4

Il donne la liste des résultats pour un paramètre biologique donné .

D. : le fichier REQUETE_BIOL

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a)

- aucun des champs "Date_entree","Espece","Race", "Sexe", "Fournisseur" de REQUETE_BIOL n'a été spécifié

- au moins un parmi les champs "Methode_analyse", "Type_echantillon","Etat_alimentaire" et "Age" a été spécifié

R. : fichier FRUIT_SELECTION_4

C. R. : ce fichier contient :
+ le numéro d'animal (num_animal)
+ le numéro de condition expérimentale (num_cond_res)
+ la date du résultat (date_res)
+ la valeur du résultat (valeur_res)
+ la phase de l'étude (etat)

j) Sous-module QTP_4_1

Il donne la liste des résultats pour un paramètre biologique donné pour les animaux pour lesquels une date d'entrée a été demandée (ceux-ci ont été sélectionnés dans un autre sous-module) et pour lesquels ni l'espèce, la race , le sexe et le fournisseur n'a été demandé . Un des critères au moins parmi la méthode d'analyse , le type d'échantillon , l'état alimentaire et l'âge a été demandé.

D. : - le fichier REQUETE_BIOL
- le fichier FRUIT_SELECTION_1

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) ainsi que FRUIT_SELECTION_1

- le champ "Date_entrée" de REQUETE_BIOL est spécifié

- aucun des champs "Espece","Race","Sexe", "Fournisseur" de REQUETE_BIOL n'a été spécifié

- au moins un parmi les champs "Methode_analyse", "Type_echantillon","Etat_alimentaire" et "Age" a été spécifié

R. : fichier FRUIT_SELECTION_4

C. R. : ce fichier est décrit au point i)

k) Sous-module QTP_4_1_FIN

Il donne la liste des résultats pour un paramètre biologique donné pour les animaux pour lesquels une date d'entrée a été demandée (ceux-ci ont été sélectionnés dans un autre sous-module) et pour lesquels aucun autre critère n'a été précisé.

D. : - le fichier REQUETE_BIOL
- le fichier FRUIT_SELECTION_1

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) ainsi que FRUIT_SELECTION_1

- le champ "Date_entrée" de REQUETE_BIOL est spécifié

- les autres champs ne sont pas spécifiés

R. : fichier FRUIT_SELECTION_6

C. R. : ce fichier contient :
+ le numéro d'animal (num_animal)
+ la date du résultat (date_res)
+ la valeur du résultat (valeur_res)
+ la phase de l'étude (etat)
+ le numéro de condition expérimentale (num_cond_res)

l) Sous-module QTP_4_2

Il donne la liste des résultats pour un paramètre biologique donné pour les animaux pour lesquels un ou plusieurs des critères parmi l'espèce, la race, le sexe a(ont) été demandé(s) (ceux-ci ont été sélectionnés par un autre sous-module). Un des critères au moins parmi la méthode d'analyse , le type d'échantillon , l'état alimentaire et l'âge a été demandé. Le fournisseur n'a pas été précisé.

D. : - le fichier REQUETE_BIOL
- le fichier FRUIT_SELECTION_2

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) et FRUIT_SELECTION_2 celle au point d)

- le champ "Date_entrée" de REQUETE_BIOL est spécifié

- le champ "Fournisseur" de REQUETE_BIOL n'est pas spécifié

- un au moins des champs "Espece","Race","Sexe", de REQUETE_BIOL a été spécifié et

FRUIT_SELECTION_2 contient le résultat de la recherche selon ces critères

- au moins un parmi les champs "Methode_analyse", "Type_echantillon","Etat_alimentaire" et "Age" a été spécifié

R. : fichier FRUIT_SELECTION_4

C. R. : ce fichier est décrit au point i)

m) Sous-module QTP_4_2_FIN

Il donne la liste des résultats pour un paramètre biologique donné pour les animaux pour lesquels au moins un des critères parmi l'espèce,la race,le sexe a été demandé (ceux-ci ont été sélectionnés dans un autre sous-module) et pour lesquels aucun des critères parmi la méthode d'analyse, le type d'échantillon , l'état alimentaire et l'âge n'a été précisé.

D. : - le fichier REQUETE_BIOL
- le fichier FRUIT_SELECTION_2

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) et FRUIT_SELECTION_2 celle au point d)

- un ou plusieurs champs parmi "Esppece","Race", "Sexe" est(sont) précisé(s) et FRUIT_SELECTION_2 contient le résultat de la recherche selon ces critères

- aucun des champs "Methode_analyse", "Type_echantillon","Etat_alimentaire","Age" et "Fournisseur" n'est spécifié

R. : fichier FRUIT_SELECTION_6

C. R. : ce fichier est décrit au point k)

n) Sous-module QTP_4_3

Il donne la liste des résultats pour un paramètre biologique donné pour les animaux pour lesquels le fournisseur a été demandé (ceux-ci ont été sélectionnés par un autre sous-module). Un des critères au moins parmi la méthode d'analyse , le type d'échantillon , l'état alimentaire et l'âge a été demandé.

D. : - le fichier REQUETE_BIOL
- le fichier FRUIT_SELECTION_3

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) et FRUIT_SELECTION_3 celle au point f)

- le champ "Fournisseur" de REQUETE_BIOL est spécifié et FRUIT_SELECTION_3 contient le résultat de la recherche selon ce critère

- au moins un parmi les champs "Methode_analyse", "Type_echantillon","Etat_alimentaire" et "Age" a été spécifié

R. : fichier FRUIT_SELECTION_4

C. R. : ce fichier est décrit au point i)

o) Sous-module QTP_4_3_FIN

Il donne la liste des résultats pour un paramètre biologique donné pour les animaux pour lesquels le fournisseur a été demandé (ceux-ci ont été sélectionnés dans un autre sous-module) et pour lesquels aucun des critères parmi la méthode d'analyse, le type d'échantillon , l'état alimentaire et l'âge n'a été précisé.

D. : - le fichier REQUETE_BIOL
- le fichier FRUIT_SELECTION_3

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) et FRUIT_SELECTION_3 celle au point f)

- le champ "Fournisseur" est spécifié et FRUIT_SELECTION_3 contient le résultat de la recherche selon ce critère

- aucun des champs "Methode_analyse", "Type_echantillon","Etat_alimentaire","Age" n'est spécifié

R. : fichier FRUIT_SELECTION_6

C. R. : ce fichier est décrit au point k)

p) Sous-module QTP_4_FIN

Il donne la liste des résultats pour un paramètre biologique donné et pour aucun autre critère.

D. : - le fichier REQUETE_BIOL

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a)

- aucun des champs deREQUETE_BIOL n'est garni (en dehors de "Nom_param_bio" d'office)

R. : fichier FRUIT_SELECTION_6

C. R. : ce fichier est décrit au point k)

r) Sous-module QTP_5_4

Il donne la liste des résultats pour un ou plusieurs critères parmi la méthode d'analyse, le type d'échantillon , l'état alimentaire en se basant sur la recherche effectuée sur le paramètre biologique. L'âge est précisé.

D. : - le fichier REQUETE_BIOL

- le fichier FRUIT_SELECTION_4

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) et FRUIT_SELECTION_4 celle au point i)

- au moins un des champs "Methode_analyse" , "Type_echantillon","Etat_alimentaire" est précisé

- FRUIT_SELECTION_4 contient le résultat de la recherche selon le paramètre biologique

- le champ "Age" est spécifié

R. : fichier FRUIT_SELECTION_5

C. R. : ce fichier contient :
+ le numéro d'animal (num_animal)
+ la date du résultat (date_res)
+ la valeur du résultat (valeur_res)
+ la phase de l'étude (etat)
+ le numéro de condition expérimentale (num_cond_res)

s) Sous-module QTP_5_4_FIN

Il donne la liste des résultats pour un ou plusieurs critères parmi la méthode d'analyse, le type d'échantillon , l'état alimentaire en se basant sur la recherche effectuée sur le paramètre biologique.L'âge n'est pas précisé.

D. : - le fichier REQUETE_BIOL

- le fichier FRUIT_SELECTION_4

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) et FRUIT_SELECTION_4 celle au point i)

- au moins un des champs "Methode_analyse" , "Type_echantillon","Etat_alimentaire" est précisé

- FRUIT_SELECTION_4 contient le résultat de la recherche selon le paramètre biologique

- le champ "Age" n'est pas spécifié

R. : fichier FRUIT_SELECTION_6

C. R. : ce fichier est décrit au point k)

t) Sous-module QTP_6_4

Il donne la liste des résultats pour des animaux pour lesquels un âge a été donné se basant sur la recherche effectuée sur le paramètre biologique sans que la méthode d'analyse, le type d'échantillon , l'état alimentaire n'aient été spécifiés .

D. : - le fichier REQUETE_BIOL

- le fichier FRUIT_SELECTION_4

C. D. : - le fichier REQUETE_BIOL a la structure décrite au point a) et FRUIT_SELECTION_4 celle au point i)

- aucun des champs "Methode_analyse" , "Type_echantillon","Etat_alimentaire" n'est précisé

- FRUIT_SELECTION_4 contient le résultat de la recherche selon le paramètre biologique

- le champ "Age" est spécifié

R. : fichier FRUIT_SELECTION_6

C. R. : ce fichier est décrit au point k)

u) Sous-module QTP_6_5

Il donne la liste des résultats pour des animaux pour lesquels un âge a été donné se basant sur la recherche effectuée sur les critères parmi la méthode d'analyse, le type d'échantillon , l'état alimentaire .

D. : - le fichier REQUETE_BIOL

- le fichier FRUIT_SELECTION_5

C. D. : - le fichier REQUETE_BIOL a la structure décrite au

point a) et FRUIT_SELECTION_5 celle au point r)

- au moins un des champs "Methode_analyse" , "Type_echantillon","Etat_alimentaire" est précisé et FRUIT_SELECTION_5 contient le résultat de la recherche selon ces critères

- le champ "Age" est spécifié

R. : fichier FRUIT_SELECTION_6

C. R. : ce fichier est décrit au point k)

v1) Sous-module TRAITEMENT_AGE

A partir des résultats présents dans FRUIT_SELECTION_6 , ce module fournit pour chaque résultat l'âge de l'animal (en jours) pour lequel ce résultat a été mesuré.

D. : le fichier FRUIT_SELECTION_6

C. D. : le fichier FRUIT_SELECTION_6 est décrit au point k)

R. : fichier texte AGE_PARAMETRE.TXT

C. R. : le fichier texte AGE_PARAMETRE.TXT contient :
+ l'âge réel en jours
+ le résultat
et est organisé de telle manière que l'âge apparait dans la première colonne et la valeur du résultat dans la deuxième colonne

REM : les résultats sont placés dans des fichiers textes car les statistiques réalisées le seront avec le logiciel RS/1 qui travaille sur de tels fichiers. C'est donc non seulement une recherche qui est effectuée mais aussi une conversion vers un fichier texte. Ceci est aussi valable pour tous les sous_modules de recherche qui vont être décrits ci-après.

v2) Sous-module TRAITEMENT_POIDS

A partir des résultats présents dans FRUIT_SELECTION_6 , ce module fournit pour chaque résultat le poids de l'animal (en grammes) pour lequel ce résultat a été mesuré.

D. : le fichier FRUIT_SELECTION_6

C. D. : le fichier FRUIT_SELECTION_6 est décrit au point k)

R. : fichier texte POIDS_PARAMETRE.TXT

C. R. : le fichier texte POIDS_PARAMETRE.TXT contient :
+ le poids en grammes (colonne 1)
+ le résultat (colonne 2)

v3) Sous-module TRAITEMENT_SEXE

A partir des résultats présents dans FRUIT_SELECTION_6 , ce module fournit pour chaque résultat le sexe de l'animal pour lequel ce résultat a été mesuré.

D. : le fichier FRUIT_SELECTION_6

C. D. : le fichier FRUIT_SELECTION_6 est décrit au point k)

R. : fichier texte SEXE_PARAMETRE.TXT

C. R. : le fichier texte SEXE_PARAMETRE.TXT contient :
+ le sexe (M ou F) (colonne 1)
+ le résultat (colonne 2)

v4) Sous-module TRAITEMENT_TYPEALIMENT

A partir des résultats présents dans FRUIT_SELECTION_6 , ce module fournit pour chaque résultat le type d'aliment reçu par l'animal pour lequel ce résultat a été mesuré.

D. : le fichier FRUIT_SELECTION_6

C. D. : le fichier FRUIT_SELECTION_6 est décrit au point k)

R. : fichier texte TYPEALIMENT_PARAMETRE.TXT

C. R. : le fichier texte TYPEALIMENT_PARAMETRE.TXT contient :
+ le type d'aliment (colonne 1)
+ le résultat (colonne 2)

v5) Sous-module TRAITEMENT_FOURNISSEUR

A partir des résultats présents dans FRUIT_SELECTION_6 , ce module fournit pour chaque résultat le fournisseur de l'animal pour lequel ce résultat a été mesuré.

D. : le fichier FRUIT_SELECTION_6

C. D. : le fichier FRUIT_SELECTION_6 est décrit au point k)

R. : fichier texte FOURNISSEUR_PARAMETRE.TXT

C. R. : le fichier texte FOURNISSEUR_PARAMETRE.TXT contient :
+ le nom du fournisseur (colonne 1)
+ le résultat (colonne 2)

v6) Sous-module TRAITEMENT_RACE

A partir des résultats présents dans FRUIT_SELECTION_6 , ce module fournit pour chaque résultat le race de l'animal pour lequel ce résultat a été mesuré.

D. : le fichier FRUIT_SELECTION_6

C. D. : le fichier FRUIT_SELECTION_6 est décrit au point k)

R. : fichier texte RACE_PARAMETRE.TXT

C. R. : le fichier texte RACE_PARAMETRE.TXT contient :
+ la race (colonne 1)
+ le résultat (colonne 2)

v7) Sous-module TRAITEMENT_ESPECE

A partir des résultats présents dans FRUIT_SELECTION_6 , ce module fournit pour chaque résultat l'espèce de l'animal pour lequel ce résultat a été mesuré.

D. : le fichier FRUIT_SELECTION_6

C. D. : le fichier FRUIT_SELECTION_6 est décrit au point k)

R. : fichier texte ESPECE_PARAMETRE.TXT

C. R. : le fichier texte ESPECE_PARAMETRE.TXT contient :
+ l'espèce (colonne 1)
+ le résultat (colonne 2)

v7) Sous-module TRAITEMENT_TYPE_INJECTION

A partir des résultats présents dans FRUIT_SELECTION_6 , ce module fournit pour chaque résultat le tupe d'injection subi par l'animal pour lequel ce résultat a été mesuré.

D. : le fichier FRUIT_SELECTION_6

C. D. : le fichier FRUIT_SELECTION_6 est décrit au point k)

R. : fichier texte TYPE_INJECTION_PARAMETRE.TXT

C. R. : le fichier texte TYPE_INJECTION_PARAMETRE.TXT
contient :
+ la voie d'injection (colonne 1)
+ le résultat (colonne 2)

w1) Sous-module EVOLUTION_POIDS

Pour une étude donnée , ce sous-module fournit pour chaque jour de celle-ci les poids des animaux qui en font partie.

D. : le fichier EVOPOIDSETUDE

C. D. : ce fichier contient le code de l'étude demandée (code_etude)

R. : fichier texte EVOPOIDS.TXT

C. R. : le fichier texte EVOPOIDS.TXT
contient :
+ le jour de l'étude (colonne 1)
+ le poids (colonne 2)

w2) Sous-module EVOLUTION_POIDS_SEXE

Pour une étude donnée , ce sous-module fournit pour chaque jour de celle-ci les poids des animaux qui en font partie ainsi que leur sexe.

D. : le fichier EVOPOIDSETUDE

C. D. : ce fichier contient le code de l'étude demandée (code_etude)

R. : fichier texte EVOPOIDS_SEXE.TXT

C. R. : le fichier texte EVOPOIDS_SEXE.TXT
contient :
+ le jour de l'étude (colonne 1)
+ le poids (colonne 2)
+ le sexe (M ou F) (colonne 3)

w3) Sous-module EVOPOIDSAGE

Ce sous-module fournit les poids des animaux en fonction de leur âge.

D. : le fichier EVOPOIDSAGE

```
C. D. : ce fichier contient :
           + l'espèce
           + la race
           + le sexe
         et les champs "Race" et "Sexe" ne sont pas requis

R.   : fichier texte EVOPOIDSAGERES.TXT

C. R. : le fichier texte EVOPOIDSAGERES.TXT
        contient :
            + l'âge              (colonne 1)
            + le poids           (colonne 2)
```

w4) Sous-module EVOPOIDSRACE

Ce sous-module fournit les poids des animaux en fonction de leur âge et de leur race.

```
D. :  le fichier EVOPOIDSAGE

C. D. : ce fichier contient :
           + l'espèce
           + la race
           + le sexe
         et le champ "Sexe" ne sont pas requis("Race" non
         demandé)

R.   : fichier texte EVOPOIDSAGERACE.TXT

C. R. : le fichier texte EVOPOIDSAGERACE.TXT
        contient :
            + l'âge              (colonne 1)
            + le poids           (colonne 2)
            + la race            (colonne 3)
```

w5) Sous-module EVOPOIDSSEXE

Ce sous-module fournit les poids des animaux en fonction de leur âge et de leur sexe.

```
D. :  le fichier EVOPOIDSAGE

C. D. : ce fichier contient :
           + l'espèce
           + la race
           + le sexe
         et le champ "Race" ne sont pas requis( "Sexe" non
         demandé)

R.   : fichier texte EVOPOIDSAGESEXE.TXT
```

C. R. : le fichier texte EVOPOIDSAGESEXE.TXT
contient :
+ l'âge (colonne 1)
+ le poids (colonne 2)
+ le sexe (M ou F) (colonne 3)

3. Module "Statistiques de biologie clinique"

Ce module logique a été scindé en deux écrans POWERHOUSE pour l'implémentation.

a. Ecran "Menu_stat_biol_clin"

Il propose les dix critères de sélection de biologie clinique et ceux-ci sont placés dans un fichier REQUETE_BIOL dont la structure a été évoquée au paragraphe 2 du point 4.5.2 . Outre ces critères , l'opérateur pour l'âge et l'opérateur pour la date d'entrée sont demandés.

Cet écran appelle le module "Organisation de la requête" qui donne l'ordre des sous-modules de recherche à lancer et c'est à ce niveau que ceux-ci sont appelés (le module "Recherche" n'est pas appelé mais un certain nombre de ces sous-modules le sont). Les sous-modules appelables sont ceux qui ont été décrits au paragraphe 4.5.2 de a) à u) .

b. Ecran "Stat_de_bio_clin"

C'est ici qu'est demandé le type de statistique de biologie clinique que l'utilisateur désire. L'utilisateur peut choisir (actuellement) parmi :

- "AGE" : le sous-module de recherche "TRAITEMENT_AGE" est appelé
- "POIDS" : le sous-module de recherche "TRAITEMENT_POIDS" est appelé
- "SEXE" : le sous-module de recherche "TRAITEMENT_SEXE" est appelé
- "TYPE_ALIMENT" : le sous-module de recherche "TRAITEMENT_TYPEALIMENT" est appelé
- "FOURNISSEUR" : le sous-module de recherche "TRAITEMENT_FOURNISSEUR" est appelé
- "RACE" : le sous-module de recherche "TRAITEMENT_RACE" est appelé
- "ESPECE" : le sous-module de recherche "TRAITEMENT_ESPECE" est appelé
- "TYPE_INJECTION" : le sous-module de recherche "TRAITEMENT_TYPE_INJECTION" est appelé
- autre chose : le message "Ce facteur est inconnu ! " est affiché

L'appel au module "Traitements statistiques" n'est pas réalisé automatiquement dans le logiciel. En effet , nous utilisons le logiciel RS/1 qui est installé actuellement sur une autre machine (CPMIC2 : microvax). Il faut donc réaliser une connection que seul l'utilisateur pourra effectuer en donnant le nom de sa directory et son mot de passe ayant préalablement obtenu un "account" sur CPMIC2. L'écran "Stat_de_bio_clin" pourra seulement donner l'ordre de connection ("set host CPMIC2") par une commande VAX.

Le transfert des fichiers obtenus vers CPMIC2 devra être réalisé et le fichier contenant les informations de la recherche de biologie clinique est appelé INFORECHERCHE. Le module "Prepa_inforecherche" réalise la conversion de REQUETE_BIOL vers INFORECHERCHE et est décrit au point c .

c. Module "Prepa_inforecherche" : implémenté sous la forme d'un QUIZZ

D. : fichier REQUETE_BIOL

C.D. : décrit au paragraphe 2 du point 4.5.2

R. : fichier texte INFORECHERCHE.TXT

C.R. : le fichier texte INFORECHERCHE.TXT a la strucure suivante :

col 0	col 1
"Entry date "	date d'entrée
"Race "	race
"Species "	espèce
"Sex "	sexe
"Tradesman "	fournisseur
"Parameter "	paramètre biologique
"Methode "	méthode d'analyse
"Sample type "	type d'échantillon
"State "	état alimentaire
"Age "	âge (en jours)
"Stat tupe "	type de statistique
"Unit "	unité de mesure (paramètre)

4. Module "Statistiques de toxico"

celui-ci n'est pas encore implémenté.

5. Module "Traitements statistiques"

Comme il a été signalé dans la "Conception logique" , ce module est constitué de toute une série de sous-modules. Ceux-ci ont été implémentés avec le langage RPL du logiciel RS/1. Les routines RS/I utilisées seront mentionnées dans le paragraphe "Routines : ". Voyons de plus près chacun des modules en employant la terminologie RS/1 (PROCEDURE).

a. Procedure STATISTIQUES

C'est le module directeur pour les statistiques de biologie clinique et de toxicologie.

D. : - le fichier INFORECHERCHE.TXT
- le fichier RECHERCHETOXICO.TXT

C.D. : - le fichier INFORECHERCHE.TXT est décrit ci-dessus
- le fichier RECHERCHETOXICO.TXT a la strucure :

col 0	col 1
"Study number "	code de l'étude
"Species "	espèce
"Race "	race
"Sex "	sexe
"Stat type "	type de dtatistique

R. : * S'il s'agit d'une demande de biologie clinique :

+ le fichier INFORECHERCHE.TXT est converti en une table RS/1 appelée INFORECHERCHE
+ si le type de statistique de INFORECHERCHE vaut :
- "TA" : * le fichier AGE_PARAMETRE.TXT est converti en une table RS/1 appelée TRTAGE
* la procédure TRAITEMENT_AGE est appelée
- "TP" : * le fichier POIDS_PARAMETRE.TXT est converti en une table RS/1 appelée TRTPOIDS
* la procédure TRAITEMENT_POIDS est appelée
- "TS" : * le fichier SEXE_PARAMETRE.TXT est converti en une table RS/1 appelée TRTSEXE
* la procédure TRAITEMENT est appelée avec
NOMTABLE = "TRTSEXE"
NOMGRAPHE = "TRTSEXEGRAPH"
LIBELLE1 = "Do you want to test the difference between sexes ?"
LIBELLE2 = "There is a difference between the two sexes ! "
- "TAL" : * le fichier TYPEALIMENT_PARAMETRE.TXT est converti en une table RS/1 appelée TRTALI
* la procédure TRAITEMENT est appelée avec
NOMTABLE = "TRTALI"
NOMGRAPHE = "TRTALIGRAPH"
LIBELLE1 = "Do you want to test the aliment type effect ?"
LIBELLE2 = "There is a aliment type effect ! "
- "TF" : * le fichier FOURNISSEUR_PARAMETRE.TXT est converti en une table RS/1 appelée TRTFOURN
* la procédure TRAITEMENT est appelée avec
NOMTABLE = "TRTFOURN"
NOMGRAPHE = "TRTFOURNGRAPH"
LIBELLE1 = "Do you want to test the difference between tradesmen ?"
LIBELLE2 = "There is a difference between

```
                                the tradesmen ! "
    - "TR" : * le fichier RACE_PARAMETRE.TXT est converti en
               une table RS/1 appelée TRTSOUCHE
             * la procédure TRAITEMENT est appelée avec
                  NOMTABLE = "TRTSOUCHE"
                  NOMGRAPHE = "TRTSOUCHEGRAPH"
                  LIBELLE1 = "Do you want to test the
                         difference between races ?"
                  LIBELLE2 = "There is a difference between
                                the races ! "
    - "ESP" : * le fichier ESPECE_PARAMETRE.TXT est converti
                en une table RS/1 appelée TRTESPECE
              *  la procédure TRAITEMENT est appelée avec
                  NOMTABLE = "TRTESPECE"
                  NOMGRAPHE = "TRTESPECEGRAPH"
                  LIBELLE1 = "Do you want to test the
                         difference between species ?"
                  LIBELLE2 = "There is a difference between
                                the species ! "
    - "TI" : * le fichier TYPE_INJECTION_PARAMETRE.TXT est
               converti en une table RS/1 appelée
               TRTINJECTION
             * la procédure TRAITEMENT est appelée avec
                  NOMTABLE = "TRTINJECTION"
                  NOMGRAPHE = "TRTINJECTIONGRAPH"
                  LIBELLE1 = "Do you want to test the
                    difference between injection types ?"
                  LIBELLE2 = "There is a difference between
                                injection types ! "

   * S'il s'agit d'une demande de toxicologie :

+ le fichier RECHERCHETOXICO.TXT est converti en une table
  RS/1 appelée RECHERCHETOXICO
+ si le type de statisyique de RECHERCHETOXICO vaut :
    - "EPE"  : * le fichier EVOPOIDS.TXT est converti en
                 une table RS/1 appelée EVOPOIDS
               * la procédure EVO_POIDS est appelée
    - "EPES" : * le fichier EVOPOIDS_SEXE.TXT est converti
                 en une table RS/1 appelée EVOPETSEXE
               * la procédure EVO_POIDS_ETD_SEXE est
                 appelée
    - "EPA"  : * le fichier EVOPOIDSAGERES.TXT est converti
                 en une table RS/1 appelée EVOPAGE
               * la procédure EVO_POIDS_AGE est
                 appelée
    - "EPR"  : * le fichier EVOPOIDSAGERACE.TXT est
                 converti en une table RS/1 appelée
                 EVOPRACE
               * la procédure EVO_POIDS_RACE est
                 appelée
    - "EPS"  : * le fichier EVOPOIDSAGESEXE.TXT est
                 converti en une table RS/1 appelée
                 EVOPSEXE
```

* la procédure EVO_POIDS_SEXE est appelée

C.R. :

- la structure de la table TRTAGE est analogue à celle du fichier AGE_PARAMETRE.TXT décrit au point 4.5.1.B (2.v1)
- la structure de la table TRTPOIDS est analogue à celle du fichier POIDS_PARAMETRE.TXT décrit au point 4.5.1.B (2.v2)
- la structure de la table TRTSEXE est analogue à celle du fichier SEXE_PARAMETRE.TXT décrit au point 4.5.1.B(2.v3)
- la structure de la table TRTALI est analogue à celle du fichier TYPEALIMENT_PARAMETRE.TXT décrit au point 4.5.1.B(2.v4)
- la structure de la table TRTFOURN est analogue à celle du fichier FOURNISSEUR_PARAMETRE.TXT décrit au point 4.5.1.B(2.v5)
- la structure de la table TRTSOUCHE est analogue à celle du fichier RACE_PARAMETRE.TXT décrit au point 4.5.1.B(2.v6)
- la structure de la table TRTESPECE est analogue à celle du fichier ESPECE_PARAMETRE.TXT décrit au point 4.5.1.B (2.v7)
- la structure de la table TRTINJECTION est analogue à celle du fichier TYPE_INJECTION_PARAMETRE.TXT décrit au point 4.5.1.B (2.v8)
- la structure de la table EVOPOIDS est analogue à celle du fichier EVOPOIDS.TXT décrit au point 4.5.1.B (2.w1)
- la structure de la table EVOPETSEXE est analogue à celle du fichier EVOPOIDS_SEXE.TXT décrit au point 4.5.1.B (2.w2)
- la structure de la table EVOPAGE est analogue à celle du fichier EVOPOIDSAGERES.TXT décrit au point 4.5.1.B (2.w3)
- la structure de la table EVOPRACE est analogue à celle du fichier EVOPOIDSRACE.TXT décrit au point 4.5.1.B (2.w4)
- la structure de la table EVOPSEXE est analogue à celle du fichier EVOPOIDSSEXE.TXT décrit au point 4.5.1.B (2.w5)
- la structure de la table INFORECHERCHE est analogue à celle du fichier INFORECHERCHE.TXT
- la structure de la table RECHERCHETOXICO est analogue à celle du fichier RECHERCHETOXICO.TXT

b. Procedure VALEURDEREF

D. : les variables NORMALITE,MOYENNE,STD,TABLEVAL,INDICE, SAMPLENAME

C.D. : - NORMALITE vaut "NSIG" si l'échantillon pour lequel on désire les valeurs de référence est normalement distribué ; vaut "SIG" dans le cas contraire

- MOYENNE est la moyenne de l'échantillon

- STD est l'écart type de l'échantillon

- TABLEVAL est le nom de la table des données

- INDICE est le numéro de la colonne où se trouve les données concernant l'échantillon

- SAMPLENAME est le nom de cet échantillon

R. : les variables UPLIMIT , LOWLIMIT , SEUIL

C.R. : - UPLIMIT contient la valeur de référence supérieure déterminée par la méthode paramétrique exposée si NORMALITE = "NSIG" , par la méthode non paramétrique exposée si NORMALITE = "SIG" (l'agorithme utilisé ici est celui exposé au point 4.3.3 6.)

- LOWLIMIT contient la valeur de référence inférieure déterminée comme ci-dessus

- SEUIL vaut :
  + "95% " si l'utilisateur a demandé d'englober 95 % de l'échantillon
  + "99% " si l'utilisateur a demandé d'englober 99 % de l'échantillon

c. Procedure TRAITEMENT_AGE

D.: - table RS/1 TRTAGE
    - table RS/1 INFORECHERCHE

C.D.: - TRTAGE contient :
  + l'âge réel en jours (col 0)
  + le résultat (col 1)

rem.: les tables RS/1 comportent une rangée 0 et une colonne 0

- INFORECHERCHE est décrit au point 4.5.2_3_C

R.: - ajustement des poids selon un seuil demandé à l'utilisateur
- présentation à l'écran du graphe avec en abscisse l'âge et en ordonnée le résultat ainsi que les caractéristiques de la recherche de biologie clinique (critères) avec la moyenne, l'écart type, la médiane de l'échantillon.
- valeur pour un âge déterminé du paramètre biologique calculé à la demande de l'utilisateur.
- intervalles de confiance proposés

C.R.: - l'ajustement peut être obtenu par:
  + régression linéaire
  + régression non linéaire (polynôme du 2° degré)
  + régression non linéaire (polynôme du 3° degré)
  + un ajustement de l'équation a * (1-EXP(-b*X))

où a est estimé à partir du maximum des valeurs du paramètre biologique
b est initialisé à 0.1
X est un âge

L'utilisateur doit entrer un seuil d'ajustement qu'il désire atteindre. Si ce seuil n'est pas atteint, le meilleur ajustement des quatres possibilités offertes est sélectionné. Si on atteint pour un ajustement (linéaire, non linéaire,...etc.) le seuil, on s'arrête à cette étape.

- la valeur pour un âge déterminé est calculée à partir de l'équation d'ajustement.
- le graphe présenté à l'écran peut être conservé sous un nom choisi par l'utilisateur. Sinon, il sera détruit.
- il en est de même pour les statistiques (moyenne, écart type, médiane).

Routines : - $FITLINE
- $FITPOLYNOMIAL
- $FITFUNCTION
- $MEASURE
- $EVALCURVE
- $_GRAPH_CI

d. Procedure TRAITEMENT_POIDS

D.: - table RS/1 TRTPOIDS
- table RS/1 INFORECHERCHE

C.D.: - TRTPOIDS contient :
+ le poids (col 0)
+ le résultat (col 1)

rem.: les tables RS/1 comportent une rangée 0 et une colonne 0

- INFORECHERCHE est décrit au point 4.5.2_3_C

R.: - ajustement des poids selon un seuil demandé à l'utilisateur
- présentation à l'écran du graphe avec en abscisse le poids et en ordonnée le résultat ainsi que les caractéristiques de la recherche de biologie clinique (critères) avec la moyenne, l'écart type, la médiane de l'échantillon.
- valeur pour un poids déterminé du paramètre biologique calculé à la demande de l'utilisateur.
- intervalles de confiance proposés

C.R.: - l'ajustement peut être obtenu par:
+ régression linéaire
+ régression non linéaire (polynôme du 2° degré)
+ régression non linéaire (polynôme du 3° degré)

+ un ajustement de l'équation a * (1-EXP(-b*X))
où a est estimé à partir du maximum des valeurs du paramètre biologique
b est initialisé à 0.1
X est un âge

L'utilisateur doit entrer un seuil d'ajustement qu'il désire atteindre. Si ce seuil n'est pas atteint, le meilleur ajustement des quatres possibilités offertes est sélectionné. Si on atteint pour un ajustement (linéaire, non linéaire,...etc.) le seuil, on s'arrête à cette étape.

- la valeur pour un poids déterminé est calculée à partir de l'équation d'ajustement.
- le graphe présenté à l'écran peut être conservé sous un nom choisi par l'utilisateur. Sinon, il sera détruit.
- il en est de même pour les statistiques (moyenne, écart type, médiane).

Routines : - $FITLINE
- $FITPOLYNOMIAL
- $FITFUNCTION
- $MEASURE
- $EVALCURVE
- $_GRAPH_CI

e. Procedure TRAITEMENT.

D.: - les variables NOMTABLE, NOMGRAPHE, LIBELLE1, LIBELLE2
- la table RS/1 INFORECHERCHE.

C.D.: - NOMTABLE : nom de la table RS/1 où se trouvent les données à traiter.
- NOMGRAPHE : nom sous lequel le graphe sera provisoirement créé.
- LIBELLE1 et LIBELLE2 : correspondent à des messages intervenant dans la procédure qui seront affichés à l'écran et qui dépendent du facteur étudié (espèce, race, sexe,...etc.).
- la table INFORECHERCHE est décrite au point 4.5.2_3_C

R.: - test de la normalité de chacun des groupes de l'échantillon (par exemple: mâles et femelles).
- dessin des courbes des groupes normaux sur le même graphe de nom présent dans la variable NOMGRAPHE avec libellé des courbes.
- affichage à l'écran du graphe ainsi produit et des critères de recherche de INFORECHERCHE.
- affichage des statistiques à savoir :
+ moyenne
+ écart type
+ médiane

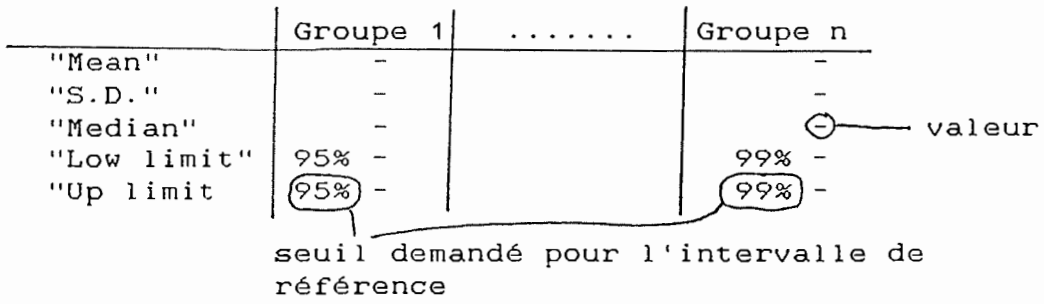
+ limite inférieure de référence
+ limite supérieure de référence

pour tous les groupes, normaux ou non.

- affichage pour chaque groupe non normal d'un histogramme des fréquences.
- test de l'égalité des moyennes.

C.R.: - les tests de normalité utilisés sont ceux présentés dans le point 3.3.2°.D
- le graphe des groupes normaux et les histogrammes relatifs à chacun des groupes anormaux peuvent être conservés sous un nom choisi par l'utilisateur.

- il en est de même pour les statistiques; la table aura la forme :



- le test d'égalité des moyennes consiste :
  * un test de la variance sur tous les groupes
  * si le test met une différence en évidence, il est offert à l'utilisateur de voir où la différence se situe. Le résultat de cette dernière opération peut être conservé dans une table sous le nom désiré par l'utilisateur.

Routines : - $TESTNORMAL
- $PROBCHI
- $HISTOGRAM
- $ANOVAONEWAY
- $SIMULTANEOUS

f. Procedure EVO_POIDS

D.: - la table RS/1 EVOPOIDS
- la table RS/1 RECHERCHETOXICO

C.D.: - la table EVOPOIDS contient :
+ le jour de l'étude (colonne 0)
+ le poids (colonne 1)
- la table RECHERCHETOXICO est décrite au point 4.5.2_5_a

R.: - demande à l'utilisateur s'il désire les intervalles de confiance

- ajustement des points selon l'équation
  a * (1-EXP(-b*X))
- affichage du graphe avec les critères de sélection de RECHERCHETOXICO et les intervalles de confiance si désirés.
- affichage de la moyenne, de l'écart type et de la médiane de l'échantillon.
- comparaison offerte des données poids concernant un animal auquel on a administré un médicament et affichage de la courbe de cet animal sur ce graphe.
- valeur pour un jour de l'étude du poids calculé à la demande de l'utilisateur.

C.R.: - le graphe présenté à l'écran peut être conservé. Sinon il sera détruit.
- pareillement pour les statistiques affichées.
- les données de l'animal servant à la comparaison peuvent provenir :
  + soit de l'utilisateur qui les introduit et qui à la fin pourra les sauver dans une table.
  + soit d'une table contenant les valeurs

Routines : -$FITFUNCTION
-$MEASURE
-$EVALCURVE

g. Procédure EVO_POIDS_ETD_SEXE

D.: - la table RS/1 EVOPETSEXE
- la table RS/1 RECHERCHETOXICO

C.D.: - la table EVOPETSEXE contient :
  + le jour de l'étude (colonne 0)
  + le poids (colonne 1)
  + le sexe (colonne 2)
- la table RECHERCHETOXICO est décrite au point 4.5.2_5_a

R.: - demande à l'utilisateur s'il désire les intervalles de confiance pour les mâles, les femelles ou les deux.
- ajustement des points selon l'équation
  a * (1-EXP(-b*X)) pour les mâles et les femelles.
- affichage du graphe avec les critères de sélection de RECHERCHETOXICO et les intervalles de confiance si désirés.
- affichage de la moyenne, de l'écart type et de la médiane de l'échantillon pour les mâles et les femelles.
- comparaison offerte des données poids concernant un animal auquel on a administré un médicament et affichage de la courbe de cet animal sur ce graphe.
- valeur pour un jour de l'étude du poids calculé à la demande de l'utilisateur pour chaque sexe.

C.R. : - le graphe présenté à l'écran peut être conservé. Sinon il sera détruit.
- pareillement pour les statistiques affichées.
- les données de l'animal servant à la comparaison peuvent provenir :
  + soit de l'utilisateur qui les introduit et qui à la fin pourra les sauver dans une table.
  + soit d'une table contenant les valeurs

Routines : -$FITFUNCTION
-$EVALCURVE

h. Procedure EVO_POIDS_AGE

D. : - la table RS/1 EVOPAGE
- la table RS/1 RECHERCHETOXICO

C.D. : - la table EVOPAGE contient :
  + l'âge en jours (colonne 0)
  + le poids (colonne 1)
- la table RECHERCHETOXICO est décrite au point 4.5.2_5_a

R. : - demande à l'utilisateur s'il désire les intervalles de confiance
- ajustement des points selon l'équation a * (1-EXP(-b*X))
- affichage du graphe avec les critères de sélection de RECHERCHETOXICO et les intervalles de confiance si désirés.
- affichage de la moyenne, de l'écart type et de la médiane de l'échantillon.
- comparaison offerte des données poids concernant un animal auquel on a administré un médicament et affichage de la courbe de cet animal sur ce graphe.
- valeur pour un jour de l'étude du poids calculé à la demande de l'utilisateur.

C.R. : - le graphe présenté à l'écran peut être conservé. Sinon il sera détruit.
- pareillement pour les statistiques affichées.
- les données de l'animal servant à la comparaison peuvent provenir :
  + soit de l'utilisateur qui les introduit et qui à la fin pourra les sauver dans une table.
  + soit d'une table contenant les valeurs

Routines : - $FITFUNCTION
- $MEASURE
- $EVALCURVE

i. Procedure EVO_POIDS_RACE

D.: - la table RS/1 EVOPRACE
    - la table RS/1 RECHERCHETOXICO

C.D.: - la table EVOPRACE contient :
        + l'âge en jours (colonne 0)
        + le poids (colonne 1)
        + la race (colonne 2)
      - la table RECHERCHETOXICO est décrite au point 4.5.2_5_a

R.: - demande à l'utilisateur s'il désire les intervalles de confiance pour les différentes races.
    - ajustement des points selon l'équation a * (1-EXP(-b*X)) pour les différentes races.
    - affichage du graphe avec les critères de sélection de RECHERCHETOXICO et les intervalles de confiance si désirés.
    - affichage de la moyenne, de l'écart type et de la médiane de l'échantillon des différents groupes.
    - comparaison offerte des données poids concernant un animal auquel on a administré un médicament et affichage de la courbe de cet animal sur ce graphe.
    - valeur pour un jour de l'étude du poids calculé à la demande de l'utilisateur pour chaque race.

C.R.: - le graphe présenté à l'écran peut être conservé. Sinon il sera détruit.
      - pareillement pour les statistiques affichées.
      - les données de l'animal servant à la comparaison peuvent provenir :
        + soit de l'utilisateur qui les introduit et qui à la fin pourra les sauver dans une table.
        + soit d'une table contenant les valeurs

Routines : - $FITFUNCTION
           - $EVALCURVE

j. Procedure EVO_POIDS_SEXE

D.: - la table RS/1 EVOPSEXE
    - la table RS/1 RECHERCHETOXICO

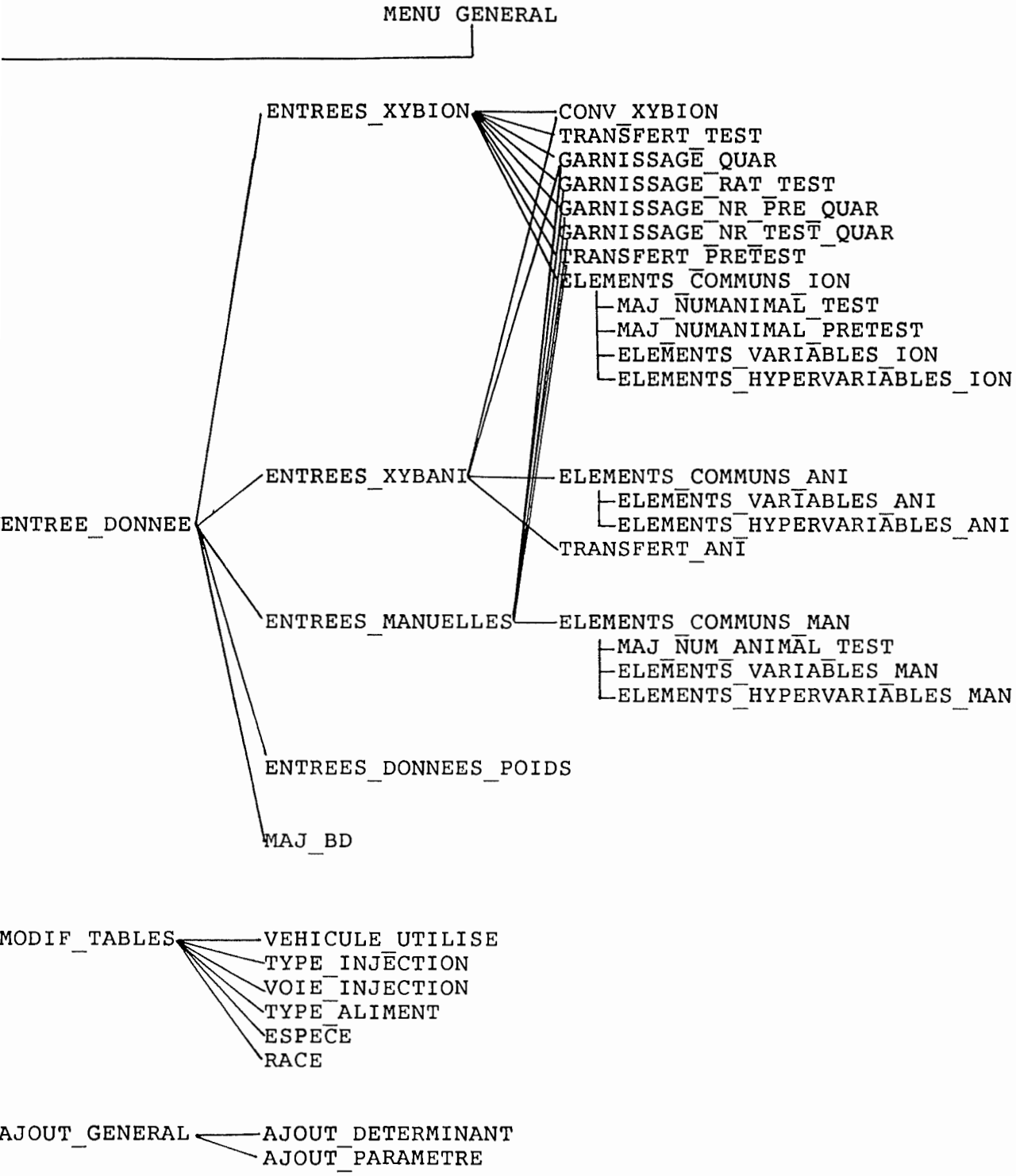
C.D.: - la table EVOPSEXE contient :
        + l'âge en jours (colonne 0)
        + le poids (colonne 1)
        + le sexe (colonne 2)
      - la table RECHERCHETOXICO est décrite au point 4.5.2_5_a

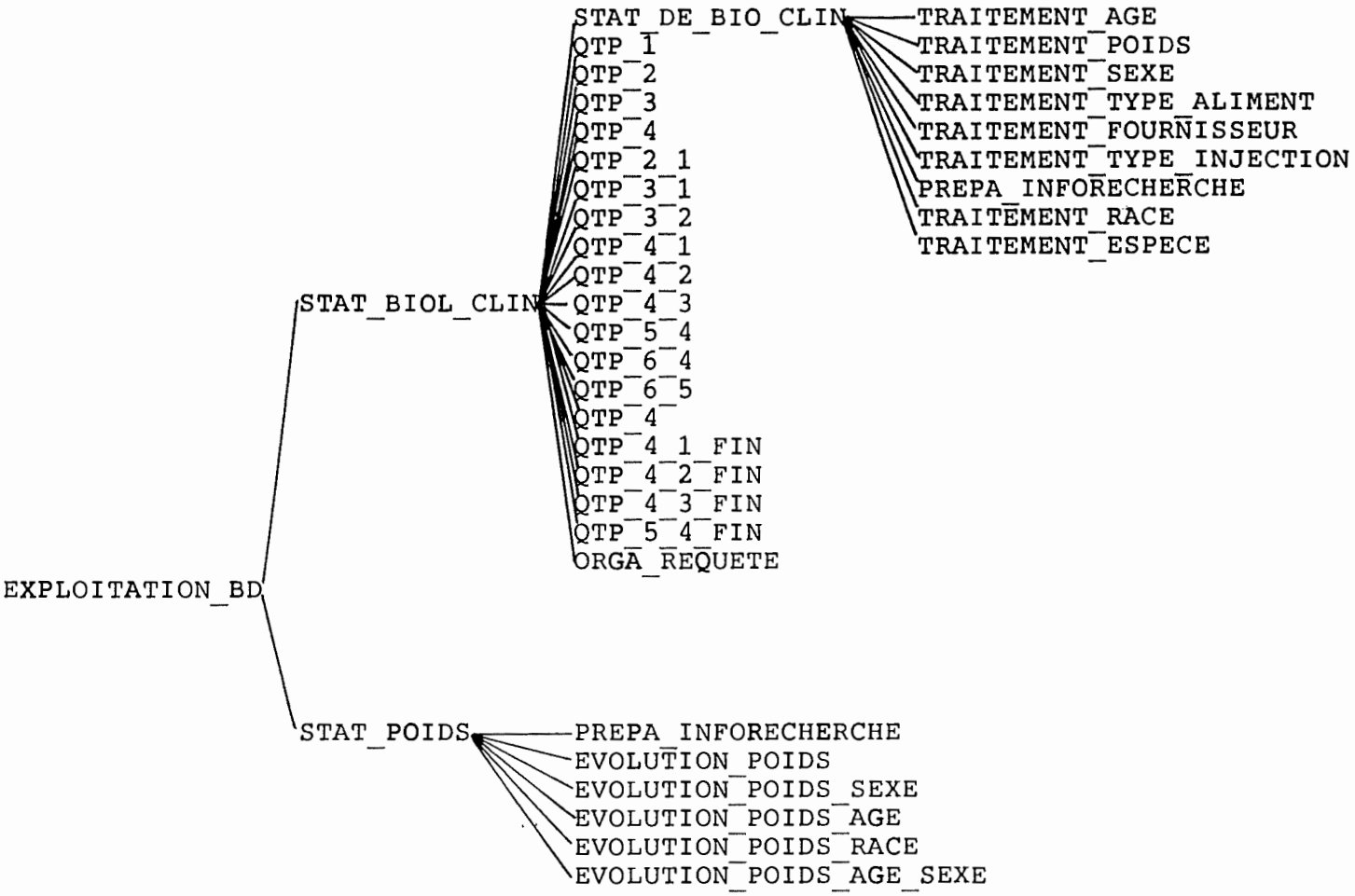
R.: - demande à l'utilisateur s'il désire les intervalles de confiance pour les différents sexes.
- ajustement des points selon l'équation a * (1-EXP(-b*X)) pour les différents sexes.
- affichage du graphe avec les critères de sélection de RECHERCHETOXICO et les intervalles de confiance si désirés.
- affichage de la moyenne, de l'écart type et de la médiane de l'échantillon des différents groupes.
- comparaison offerte des données poids concernant un animal auquel on a administré un médicament et affichage de la courbe de cet animal sur ce graphe.
- valeur pour un jour de l'étude du poids calculé à la demande de l'utilisateur pour chaque sexe.

C.R.: - le graphe présenté à l'écran peut être conservé. Sinon il sera détruit.
- pareillement pour les statistiques affichées.
- les données de l'animal servant à la comparaison peuvent provenir :
  + soit de l'utilisateur qui les introduit et qui à la fin pourra les sauver dans une table.
  + soit d'une table contenant les valeurs

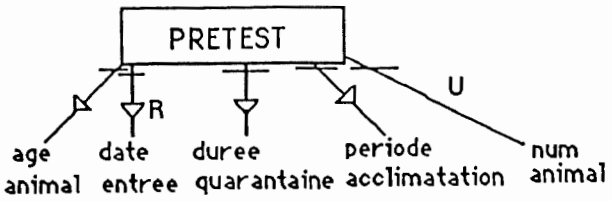
Routines : - $FITFUNCTION
- $EVALCURVE

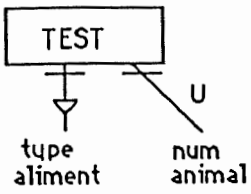
4.5.2. SCHEMA DES LIENS ENTRE MODULES

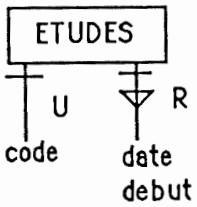


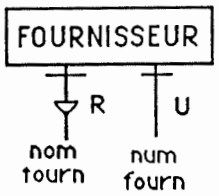

EXPLOITATION_BD
STAT_BIOL_CLIN
STAT_DE_BIO_CLIN
TRAITEMENT_AGE
TRAITEMENT_POIDS
TRAITEMENT_SEXE
TRAITEMENT_TYPE_ALIMENT
TRAITEMENT_FOURNISSEUR
TRAITEMENT_TYPE_INJECTION
PREPA_INFORECHERCHE
TRAITEMENT_RACE
TRAITEMENT_ESPECE
QTP_1
QTP_2
QTP_3
QTP_4
QTP_2_1
QTP_3_1
QTP_3_2
QTP_4_1
QTP_4_2
QTP_4_3
QTP_5_4
QTP_6_4
QTP_6_5
QTP_4
QTP_4_1_FIN
QTP_4_2_FIN
QTP_4_3_FIN
QTP_5_4_FIN
ORGA_REQUETE
STAT_POIDS
PREPA_INFORECHERCHE
EVOLUTION_POIDS
EVOLUTION_POIDS_SEXE
EVOLUTION_POIDS_AGE
EVOLUTION_POIDS_RACE
EVOLUTION_POIDS_AGE_SEXE

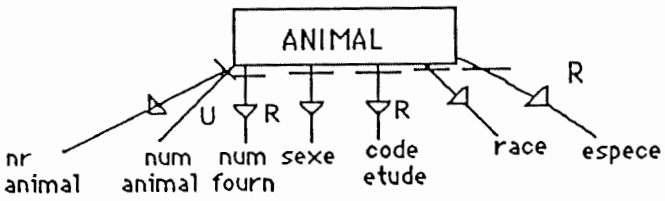
4.5.3. ENSEMBLE DES FICHIERS DE LA BASE DE DONNEES.

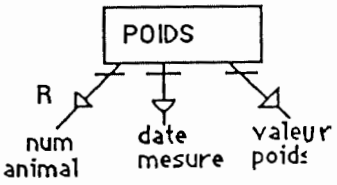


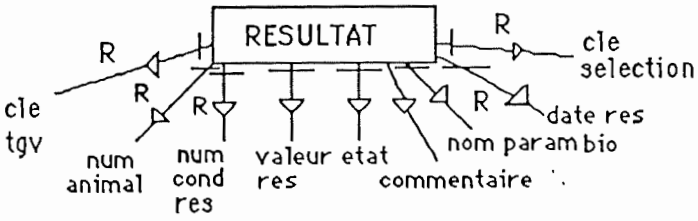


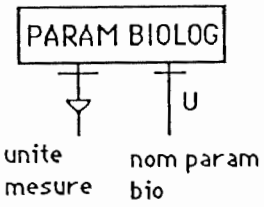


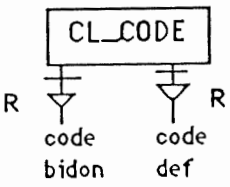


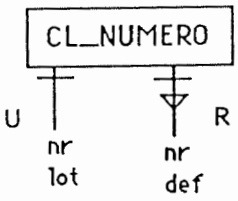


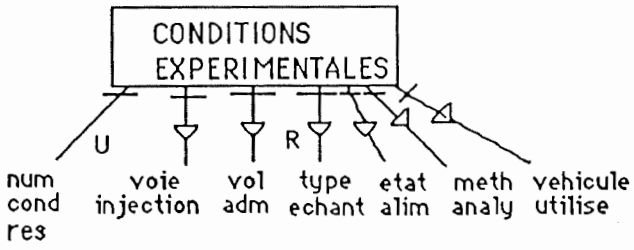


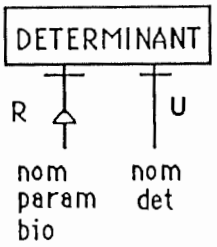












Légende : U = clé d'accès unique
R = clé d'accès répétée

CHAPITRE 5 : PERSPECTIVES ET EVOLUTION DU LOGICIEL.

5. PERSPECTIVES ET EVOLUTION DU LOGICIEL.

Au terme de la réalisation du logiciel , il reste à implémenter certains écrans concernant les statistiques de toxicologie ainsi que certains QUIZ de recherche. Ainsi, le module "Statistiques de toxicologie" permettant de sélectionner le type de statistique de toxicologie que l'on désire effectuer et les rapports provenant de la sélection de données de toxicologie correspondantes ne sont pas encore faits.

Les problèmes de sécurité au point de vue utilisateur (droits d'accès,...etc) n'ont pas été abordé ainsi que le problème des fichiers audit (qui permettent de garder une trace de toute modification ou rejets des données provenant des logiciels XYBION et XYBANI).

Une fois ces points résolus , le logiciel pourra être mis en production.

Le logiciel est conçu de telle manière que de nouvelles statistiques pourront être implémentées facilement. Par exemple, il serait aisé, grâce à la séparation existante entre la recherche d'information suivant les dix critères de biologie clinique déjà cités et les traitements statistiques proprement dits, d'ajouter de nouveaux modules spécifiques à d'autres types de traitement statistique.

En outre, on peut envisager que les données proviennent d'autre part que de XYBION/XYBANI. Il suffit dans ce cas de prévoir une conversion des nouvelles structures de données en notre structure propre (FICH_CONVERTI).
A ce propos, l'entreprise a dernièrement acquis un appareil de mesures stockant lui-même des données. Il sera intéressant de pouvoir récupérer celles-ci pour les transférer dans la B.D.

CHAPITRE 6 : DISCUSSION SUR L'UTILISATION PRATIQUE D'UN LANGAGE DE QUATRIEME GENERATION.

6. DISCUSSION SUR L'UTILISATION PRATIQUE D'UN LANGAGE DE QUATRIEME GENERATION.

Dans ce point , nous nous proposons d'établir une comparaison , basée sur les problèmes que nous avons rencontrés en pratique , entre le langage POWERHOUSE et le langage PASCAL (en tant que langage de troisième génération). Cette comparaison ne se prétend pas exhaustive et veut seulement montrer quelques difficultés et avantages de l'emploi d'un L4G.

Nous allons voir les principales étapes du développement d'un projet et signaler les différences existant entre les deux langages.

Les spécifications

Elles doivent être moins techniques avec POWERHOUSE qu'avec le PASCAL. Les arguments , leur format , doivent être clairement définis en PASCAL alors que grâce au dictionnaire de POWERHOUSE , ce fardeau est allégé. On pourra donc s'attacher davantage aux fonctionnalités à atteindre.

En conséquence , le volume des spécifications est plus faible (moins de détails).

La conception

Contrairement au PASCAL , POWERHOUSE est un langage non algorithmique. Pour ce premier , il est fondemental de concevoir les algorithmes avant le codage ce qui n'est pas toujours le cas avec POWERHOUSE. Il est en effet possible de concevoir tout en programmant vu le caractère non procédural de ce dernier.

Cependant , pour des opérations plus délicates , il est nécessaire d'utiliser des procédures offertes par POWERHOUSE et dans ce cas , il est clair que l'on retrouve le côté algorithmique (c'est le cas par exemple dans les écrans d'entrée de données où une recherche est effectuée pour voir si l'animal entré n'est pas déjà dans la B.D. et auquel cas , certaines de ses caractéristiques tels l'âge au moment de l'entrée en quarantaine , la durée de celle-ci,etc... peuvent être recherchées). Il en est de même pour des opérations complexes d'accès , de mise à jour et de suppression dans la B.D.

La programmation (codage)

* Avantages de POWERHOUSE par rapport au PASCAL

- le dictionnaire évite de devoir redéclarer les variables , les strucures des fichiers à chaque fois que celles-ci sont utilisées. Il permet aussi une standardisation des noms.

- les instructions POWERHOUSE sont de plus haut niveau , plus générales ceci facilitant des opérations de lecture , d'accès à des fichiers ... Ainsi , par exemple , si l' on désire vérifier si un nom introduit par l' utilisateur correspond à un nom valide présent dans un fichier déterminé, on peut le faire avec un LOOKUP (NOT) ON sur le fichier. Il faut signaler cependant que ce LOOKUP se réfère à un champ qui est une clé d'accès (au sens POWERHOUSE) indexée.

- la conséquence des deux avantages évoqués ci-dessus est que le nombre de lignes de code est diminué d'autant (les déclarations occupant souvent une bonne part du code en PASCAL ou en COBOL surtout !).

- la facilité pour créer des écrans avec POWERHOUSE est accrue.

- enfin , il faut signaler l'aspect plus convivial et plus interactif de ce dernier.

* Inconvénients de POWERHOUSE par rapport au PASCAL

- l'aspect "routine réutilisable" est perdu par rapport à PASCAL. C'est plutôt un enchainement de modules qui se base sur l'étape précédente. Il y a donc un aspect de flux d'informations , de séquence. Ceci est particulièrement vrai avec les QTP et les QUIZ. En PASCAL , une procédure peut être appelée autant de fois qu' on le désire ce qui n'est pas le cas en POWERHOUSE.

- le caractère d'indépendance entre certaines requêtes (QTP) est lié au point précédent. Un QTP est découpé en REQUEST et les variables d'une REQUEST sont locales. Il existe bien des variables globales mais cela pose parfois certains problèmes.

- lorsque la complexité du problème est élevée , il devient difficile de s'en sortir avec POWERHOUSE d'où la nécessité d'avoir la possibilité d'utiliser des routines PASCAL. C'était notemment le cas avec la conversion de fichiers extraits de XYBION / XYBANI (cfr Conception du logiciel).

- avec POWERHOUSE , on a tendance à raisonner avec des fichiers. Il n'existe pas de possibilité de travailler sur des tableaux en mémoire centrale. Ceci peut avoir des conséquences sur les performances. Il faut également signaler que les booléens n'existent pas en tant que tels.

- la diminution du nombre de lignes de code n'est pas toujours très importante : il est difficile de réutiliser certains écrans ou certains QTP.

- les recherches imbriquées sur des fichiers différents ne sont pas possibles avec un écran POWERHOUSE. Une recherche dans un fichier est assurée par un "WHILE RETRIEVING fichier" et ne peut contenir une autre instruction du même type. Cette imbrication est pourtant d'usage fréquent.

L'approche de programmation d'un L4G est fort différente et particulière. Elle surprend voir même laisse perplexe ceux qui ont déjà programmé dans un langage comme le PASCAL. La compréhension de ce qui est réalisé est parfois moins bonne. En PASCAL , on voit toutes les opérations exactes qui sont réalisées (bien que par rapport à l'assembleur , il y a aussi des opérations de plus haut niveau !). Il faut cependant reconnaître les qualités indiscutables d'un L4G.

Les tests

Beaucoup de tests plus techniques sont supprimés au profit de tests essentiellement logiques. C'est une conséquence même de la nature du langage POWERHOUSE. De plus, pour les écrans , l'implémentation peut se faire par essais et erreurs car le résultat peut être rapidement visualisé. Déjà à la compilation , la structure de l'écran est présentée permettant d'éviter l'éxécution de celui-ci.

Cependant , il est parfois plus difficile de voir ou se situe l'erreur. En PASCAL , la logique est entièrement développée par le programmeur et celui-ci a la possibilité de placer des compteurs ou d'autres moyens de localisation d'une erreur. Ce n'est pas le cas avec POWERHOUSE. Ne connaissant pas toujours la logique interne des instructions , des erreurs peuvent subvenir et elles sont souvent difficiles à trouver. A titre d'exemple , dans les QTP , les accès à des fichiers peuvent être réalisés via l'option "LINK". Comme il a déjà été expliqué au point 3.2 , le "LINK" permet d'accéder un fichier suivant une clé en utilisant le champ d'un premier fichier :

```
ACCESS  fichier1  LINK  fichier2
            (AND)  LINK  fichier3  LINK champ2 du fichier3
                                   TO   champ1 du fichier4
```

Cette opération a différentes modalités présentées ci-dessus. L'utilisation d'une modalité non adéquate peut mener à des résultats tout à fait différents mais la difficulté réside dans le fait qu'on ne connait pas la manière exacte dont l'opération est réalisée.

Par exemple , si aucun champ du fichier de départ n'est précisé , POWERHOUSE essaie de trouver un champ de même nom dans les deux fichiers sur les quels s'effectue le "LINK". Un oubli de préciser le champ peut entrainer une grande différence entre ce qu'on voulait faire et ce qui est réellement fait ! L'option "AND" permet d'effectuer des "LINK" en parallèle et là aussi , il faut bien faire attention à ce qu'on désire faire. De telles erreurs peuvent survenir d'autant que trois , quatre "LINK" sont parfois mis en cascade !

La maintenance

Elle est d'une manière générale plus commode avec POWERHOUSE. Il y a essentiellement deux raisons à cela :

+ les changements sont plus facile à localiser . Par exemple , si une clé ou un champ d'un fichier doivent être modifiés , celui-ci ne devra être changé que dans le dictionnaire (avec recompilations d'écrans , de QTP, de QUIZ ou le fichier est déclaré). En PASCAL , partout où le fichier est déclaré , il faut apporter les modifications désirées ce qui peut être long et difficile à localiser.

+ la logique des instructions POWERHOUSE étant plus proche de la logique humaine , ou tout au moins plus fonctionnelle , le changement à apporter est plus rapidement perçu.

Elle est également plus rapide avec POWERHOUSE qu'en PASCAL ce qui est une conséquence de ce qui vient d'être dit à l'instant et du fait que les instructions sont de plus haut niveau (donc moins nombreuses).

L'apprentissage

Contrairement à ce que l'on imagine , il y a davantage à apprendre avec le langage POWERHOUSE qu'avec le PASCAL . Une fois que les structures de base de celui-ci sont comprises (affectation,condition,WHILE,FOR,REPEAT,etc...) , il est déjà possible de programmer presque n'importe quel algorithme. Par contre , si l'on désire réaliser la même chose avec POWERHOUSE , il faut acquérir une maîtrise plus poussée du langage. Il y a en effet plus de possibilités , de particularités qui une fois maîtrisées permettent de gagner du temps.

La vocation de facilité d'apprentissage que doit posséder un L4G n'est donc pas tout à fait remplie par POWERHOUSE.

En conclusion , malgré les inconvénients qui ont été signalés dans cette comparaison , POWERHOUSE est un outil assez puissant pour la conception d'applications. L'approche suivie par ce langage convient particulièrement bien pour des niveaux de complexité moyens ou faibles mais nécessite pour des applications plus complexes des possibilités d'utilisation de langages de troisième génération.

CHAPITRE 7 : C O N C L U S I O N S

7. CONCLUSION

Ce mémoire nous a permis de nous familiariser avec un environnement de travail totalement différent de celui que nous connaissions aux Facultés. Nous avons ainsi été confronté à toute une série de problèmes parfois très pratiques. Il a fallu préciser avec les utilisateurs eux-mêmes ce qu'ils attendaient du logiciel, ceux-ci ne sachant pas toujours ce qu'ils veulent et comment le réaliser. Plusieurs réunions ont été nécessaires pour clarifier cela.

D'autre part, nous avons dû maîtriser un nouveau type de langage de programmation (Powerhouse) dont l'approche est assez différente de celle que nous avons vis-à-vis du Pascal ou du Cobol. De même, pour la partie statistique, l'apprentissage du logiciel RS/1 a été nécessitée par le choix de ce package existant sur place et auxquels les gens sont habitués.

Tout ceci nous a cependant permis d'acquérir des connaissances pratiques qui nous seront sans doute très utiles plus tard. Nous avons l'expérience d'une approche langage de quatrième génération qui dans l'avenir deviendra prépondérante dans le secteur de l'informatique de gestion.

De plus, nous avons eu la chance de voir comment l'informatique est utilisée dans une entreprise et comment elle doit être un moyen de satisfaire au mieux les besoins des utilisateurs.

CHAPITRE 8 : R E F E R E N C E S

R E F E R E N C E S

ALBERT 83 , ALBERT A.,GUEGUEN R.,SACHS Ch. "Traitement des valeurs de référence et détermination de l'intervalle de référence."
ANN.BIOL.CLIN. 41 63-79 1983.

ASH 80 , ASH K.O. "Reference intervals(normal ranges): a challenge to laboratorians."
AM.J.MED.TECHNOL. 46:504 1980.

BARNETT 78 , BARNETT R.N. "The medical usefullness of stat test."
AM.J.CLIN.PATHOL. 69:520 1978.

BARR 81 , BARR J.T. "A priori considerations when using laboratory determinations in kosteffectiveness and clinical decision analysis."
AM.J.MED.TECHNOL. 47:619 1981.

BBN 88 , BBN 88 "RS/1 user's guide."
Software Product Corporation 1988.

BODART 88 , BODART F. "Conception de systèmes d'information."
Notes de cours 1988.

BOYD 83 , BOYD J.C. ET AL. "Medical education in laboratory testing : an approach in corporating the students own laboratory results."
AM.J.CLIN.PATHOL. 79:211 1983.

COGNOS 89 , COGNOS S.A.
"Powerhouse User's manual."
1989.

CP 89 , CONTINENTAL PHARMA "Interpretation of Laboratory test data."
Article interne (J.DOUMONT).

DAGNELIE 70 , DAGNELIE P."Théorie et méthodes statistiques I"
Les presses agronomiques de Gembloux,vol 1 1970

DAGNELIE 73 , DAGNELIE P. "Théorie et méthodes statistiques II"
Les presses agronomiques de Gembloux,vol 2 1973

DAWSON 80 , DAWSON D.V. et al. "Confidence bands for the growth of head circomference in achondroplastic children during the first year of life." AMERICAN JOURNAL of MEDICAL GENETICS vol. 7 529-536 1980.

DYBKAER 87 , DYBKAER R. "Representation of observed values related to reference values." Approved recommendation(1987) on the theory of Reference values J.CLIN.CHEM.CLIN.BIOCHEM. vol. 25 657-662 1987.

FEYTMANS 76 , "Introduction à la statistique" FNDP-NAMUR , Unité de biologie quantitative 1976

FEYTMANS 86 , "Biostatistique" FNDP-NAMUR , Unité de biologie quantitative 1986

GALEN 79, GALEN R.S. "New math in the lab : predictive value theory." DIAGN.MED. 2(1):23 1979.

GALEN 80, GALEN R.S. "New math in the lab : notions of normality." DIAGN.MED. 3(4):59 1980.

HAINAUT 86 , HAINAUT J.L. "Conception assistée des applications informatiques: 2-Conception de la base de données." Presses universitaires de NAMUR MASSON 1986.

HENNEAUX 87 , HENNEAUX T.,SMETS. "Les outils de quatrième génération" BUMP-NAMUR 1987.

LUM 83 , LUM.G,BELER M.F. "Reference groups : comparing oranges with oranges and apples with apples." JAMA 249:1890 1983.

MARTIN 83 , MARTIN J. "L'informatique sans programmeurs" Les éditions d'organisation 1983 .

MARTIN 85 , MARTIN J. "Fourth generation languages." PRENTICE HALL ENGLEWOOD CLIFF (N.J.) vol.1 1985-1986

NAUS 80 , NAUS A.J. et al. "The use of patient data for the calculation of reference values for some haematological parameters." J.CLIN.CHEM.CLIN.BIOCHEM. vol 18 621-625 1980.

NOIRHOMME 89 , NOIRHOMME M. "Statistiques appliquées"
Notes de cours 1989.

RIEGELMAN 80 , RIEGELMAN R.K. : "The limits of the
laboratory : nuances of normality."
POSTGRAD MED. 70:203 1980.

ROSSING 79 , ROSSING R.G.,HATCHER W.E. : "Percentile as
reference values for laboratory data."
AM.J.CLIN.PATHOL. 72:94 1979.

SHULTZ 85 , SHULTZ E.K. "Improved reference-interval
estimation."
CLINICAL CHEMISTRY vol.31 No.12 1974-1978 1985.

SOLBERG 87a , SOLBERG H.E. "The concept of reference values"
Approved recommendation(1986) on the theory of
Reference values
J.CLIN.CHEM.CLIN.BIOCHEM.
vol. 25 337-342 1987.

SOLBERG 87b , SOLBERG H.E. "Statistical treatment of
collected reference values.Determination of
reference limits."
Approved recommendation(1987) on the theory of
Reference values
J.CLIN.CHEM.CLIN.BIOCHEM.
vol. 25 645-656 1987.

SUNDERMAN 75 , SUNDERMAN F.W. "Current concepts of 'Normal
values','Reference values' and 'discrimination
values' in clinical chemestry."
CLIN.CHEM. 21:1873 1975.

TANGO 81 , TANGO T. "An interpretation of normal ranges based
on a new concept 'individual difference quotient'
of clinical laboratory data."
MED.INFORM. vol.6 No.3 161-174 1981.

VAN LAMSWEERDE 88 , VAN LAMSWEERDE A. "Methodologie de
développement de logiciels."
Notes de cours 1988.

A N N E X E S

A N N E X E 1

CONTINENTAL PHARMA INC. - Mont-Saint-Guibert **MEMO**
From : Olivier Closson

Date : 8 novembre 1988 Cc : J.P. Bernard
A. Leroy
M. Degreef
G. Healey

Subject : Cahier de charges

Ref. : OGC/md/880005

To : P. Debras
E. Debruyne
J. de Gerlache
S. Denoel
J. Doumont
M. Masson
J. Roba

Vous trouverez en annexe la première édition du cahier des charges (réf. 87070/CC) relatif à la création d'une base de données historique de paramètres biologiques.

Nous attendons vos suggestions et commentaires pour le vendredi 25 novembre au plus tard.

Nous restons à votre entière disposition pour discuter du contenu de ce cahier de charges.

Bien à vous.

O. Closson

RESEARCH INFORMATION SYSTEMS

APPLICATIONS

****

BASE DE DONNEES HISTORIQUE DES PARAMETRES BIOLOGIQUES

****

C A H I E R D E S C H A R G E S

87070/CC

N° Edition : 1

Auteur(s): Vincent MARCHANT
François-Xavier FRIES

N° Version : 1.01

Date : 8 novembre 1988

APERCU GLOBAL DU TRAVAIL

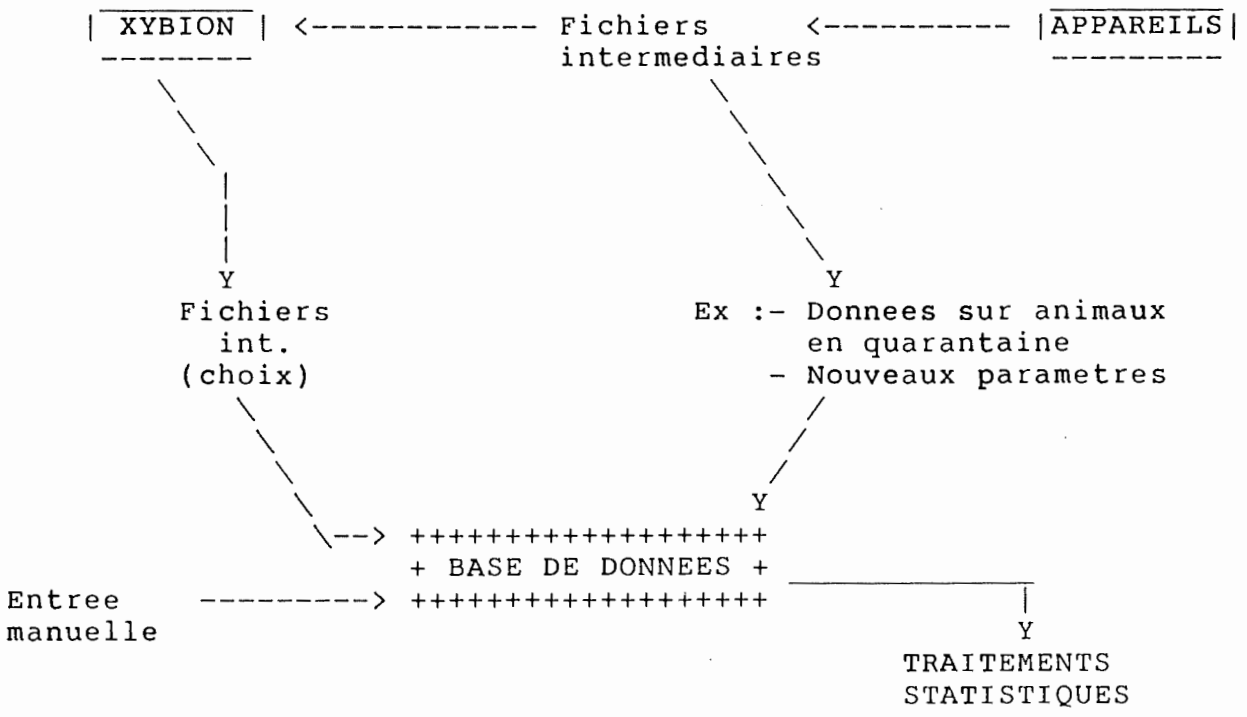
But du projet

Ce travail consiste à concevoir une "base de données" historique (en réalité, celle-ci ne sera pas basée sur un Système de Gestion de Base de Données mais sur une manipulation de fichiers RMS) peuplée, entre autres, de paramètres biologiques . Ceux-ci portent sur des animaux faisant partie des groupes de contrôle lors d'études toxicologiques ou sur des animaux en quarantaine qui ne font donc pas partie de telles études.
La manipulation de cette base de données permettra d'extraire des valeurs de ces groupes témoins afin d'essayer d'établir des distributions normales de ces paramètres.

Provenance des données

Les données proviennent :
1) du logiciel XYBION, en grande partie.
2) des résultats provenant directement des appareils de mesure
   ex : - dans le cas des animaux en quarantaine
        - nouveaux paramètres
3) des résultats introduits manuellement.

Transferts vers la base de données



A) Transfert de Xybion vers la BD

Le logiciel XYBION utilisant des formats de fichiers qui lui sont propres, il est nécessaire de convertir ces fichiers pour pouvoir manipuler aisément leur contenu.
Dans ce but, plusieurs possibilités s'offrent à nous.
La première serait d'utiliser les extractions standards offertes par XYBION qui portent sur un seul paramètre à la fois.
La seconde serait de créer un interface entre XYBION et la base de données. Enfin, il existe un logiciel offrant un transfert par groupe de paramètres.
C'est à ce niveau que l'on doit offrir à l'utilisateur la possibilité de rejeter certains résultats ne pouvant pas être pris en compte ; ceci devant être laissé à l'appréciation du chercheur. Les raisons du rejet devront être conservées (dans un fichier archives) pour des justifications ultérieures.

ex : - si l'on dispose de plusieurs mesures concernant un même animal effectuées à différents moments trop proches les uns des autres, il faut éviter de les prendre toutes en considération ce qui pourrait fausser la distribution.

- il faut éviter de peupler plusieurs fois la BD avec des données déjà introduites (ceci est vrai pour quelqu'entrée que ce soit dans la BD).

De plus, afin de pouvoir rajouter des informations non présentes dans XYBION, il paraît intéressant d'offrir la possibilité d'introduire d'autres données. Ceci peut être réalisé en offrant un écran de saisie à l'utilisateur qui ne reprendrait que les nouveaux types d'information.

B) Transfert des appareils vers la BD

Par exemple, les données recueillies au stade de la mise en quarantaine n'ont pas été transférées dans XYBION, mais néanmoins ont été stockées. Présentes dans la BD, elles pourraient faire l'objet d'études.
Ce transfert pourrait être aussi nécessaire dans le cas où les appareils de mesure fourniraient de nouveaux paramètres.

C) Entrée manuelle des résultats

On peut, par exemple, citer le cas où les données expérimentales ne seraient pas enregistrées du fait d'une défaillance lors des transferts des appareils vers Xybion, ce qui nécessiterait l'entrée manuelle des données.
En outre, lors d'études, on utilise parfois des échantillons pour la mise au point d'analyses ; ceux-çi proviennent d'animaux en quarantaine, bien identifiés.
Les valeurs obtenues avec ces échantillons sont susceptibles de participer à l'ensemble des données qui permettront d'établir une distribution.
Ceci est particulièrement vrai dans le cas des singes pour lesquels on disposera de peu de valeurs.

Traitement des données.

A partir des informations stockées dans la base de données, un certain nombre de traitements statistiques seront envisagés.
Deux logiciels statistiques sont disponibles pour cela : le logiciel SAS et le logiciel RS1. Le choix de l'un ou l'autre sera discuté avec le statisticien.
La base de données risquant de devenir énorme au fil des années, il serait intéressant de résumer les résultats d'une année par exemple. Il serait alors impossible de prendre en compte les valeurs individuelles d'un animal.

Choix des paramètres à stocker.

Une première discussion avec les utilisateurs potentiels nous permet de sélectionner les paramètres suivants :

- paramètres de sélection :
  - Espèce,
  - Age,
  - Race (souche),
  - Provenance (fournisseur),
  - Environnement (litière,cage),
  - Sexe,
  - Voie d'injection,
  - Volume administré,
  - Véhicule utilisé,
  - ~~Numéro d'opérateur,~~
  - Méthode d'analyse,
  - Code de l'étude,
  - Durée période de prétest,
  - Période d'acclimatation,
  - Date entrée dans l'animalerie,
  - Etat alimentaire (à jeun ou non),
  - Type d'échantillon (serum, plasma ou urine).
  - Type d'aliment reçu

- paramètres de biologie clinique :
  une liste se trouve en fin de rapport.

- paramètres de croissance des animaux :
  poids corporel,
  gain de poids.

- données "in life" :
  (consommation alimentaire,
  consommation de boisson.)

- commentaires significatifs.

A titre d'exemple, pour la biologie clinique, les critères suivants seraient demandés :

1. Espèce
2. Souche
3. Fournisseur
4. Sexe
5. Age
6. Date entrée dans l'animalerie
7. A jeun ou non
8. Type d'échantillon

Ensuite, on demanderait le nom du paramètre biologique sur lequel porterait la recherche ainsi que la méthode d'analyse utilisée.

Rem: la méthode d'analyse doit absolument être considérée car les valeurs normales peuvent changer suivant le type de méthode utilisée.
Par ailleurs, ce sera l'utilisateur qui sera responsable de la classification des méthodes. Un changement de température, par exemple, pour une certaine méthode devra être considéré par l'utilisateur comme une méthode différente.

Un exemple de type de demande :

quel est le taux de sucre chez le singe, de telle race, provenant du fournisseur Y, mâle , de 4 ans, introduit dans l'animalerie au cours de l'année 1987-1988, à jeun, dans du plasma ?

N.B. : 1. Certaines valeurs de paramètres (âge par exemple) n'étant pas reprises dans XYBION, pourront être entrées ici.

2. Le logiciel offert doit pouvoir fournir la possibilité de considérer d'autres paramètres que ceux déjà utilisés.
Cela assurera une souplesse plus grande en vue de nouveaux désirs de la part des utilisateurs.

A N N E X E 2

RESEARCH INFORMATION SYSTEMS

APPLICATIONS

****

BASE DE DONNEES HISTORIQUE

DE PARAMETRES BIOLOGIQUES

****

R A P P O R T D E F A I S A B I L I T E

Ref. 87070/RF

N° Edition : 1

N° Version : 1.01

Date : 6 mars 1989

Auteurs : Vincent Marchant
François-Xavier Fries

INTRODUCTION

Ce rapport est destiné à préciser l'élaboration de la base de données de paramètres biologiques. Il fait suite au cahier des charges de référence OGC/md/880005.
Nous y aborderons d'abord à partir de quoi et comment nous garnirons la BD pour parler ensuite de l'exploitation de celle-ci.

ANALYSE

1. Provenance des données

Les données proviendront d'une part du logiciel XYBION et d'autre part, contrairement à ce qui a été dit précédemment, de XYBANI. Celui-ci est identique à XYBION du point de vue logiciel mais contient cependant les données des animaux en quarantaine.

De plus l'entrée de données pourra se faire manuellement .

a. A partir de XYBION :

Le prélèvement des données de XYBION se fera par une extraction selon le code d'une étude pour un déterminant donné (càd pour un ou plusieurs paramètres).
Les informations obtenues seront :

- nom du déterminant,

- pour chaque paramètre :

    * nom du paramètre
    * numéro d'animal
    * sexe de l'animal
    * groupe
    * sous-groupe
    * jour de l'étude
    * valeur du résultat.

Le numéro d'animal a été attribué en début d'étude et identifie un animal.

En ce qui concerne le groupe, étant donné que nous ne conservons que les résultats de groupes témoins, le numéro de groupe à considérer sera toujours le même.
De plus, nous ne tiendrons pas compte du sous-groupe étant donné qu'il n'est pas utilisé.

b. A partir de XYBANI :

Les modalités de prélèvement des données sont les mêmes que pour XYBION. Seule l'interprétation des informations obtenues diffèrera :

- le code-étude a été attribué sans avoir ici la signification qu'il a lors d'une étude réelle;
- le numéro d'animal consiste en fait à associer le numéro de lot avec un numéro séquentiel;
- le jour de l'étude fera référence à la date d'entrée en quarantaine de l'animal;
- le groupe a toujours la même valeur puisqu'il ne peut s'agir que de témoins.

c. Conversion des extractions :

Nous avons décidé de travailler sur un fichier constitué à partir de celui provenant d'une extraction de XYBION / XYBANI et dont le format nous est propre car :

- le fichier provenant d'une extraction n'est pas utilisable tel quel; une conversion du fichier ASCII de XYBION en un fichier aisément manipulable par POWERHOUSE doit être réalisée;

- cela nous permet de rester indépendant de l'origine des données à considérer. Par exemple, dans le cas où on disposerait d'une nouvelle version de XYBION, il suffirait de prévoir une conversion du nouveau type de fichier vers notre format standard.

2. Analyse du déroulement de l'entrée des données

2.1. Typologie des entrées

Pour éviter à l'utilisateur d'entrer plusieurs fois des éléments concernant tous les animaux d'une étude, nous avons décidé de les lui faire entrer via un écran de saisie une seule fois (on parlera d'éléments communs).

Nous avons opéré de la même manière pour les caractéristiques d'un animal.
En effet, à un animal peut correspondre plusieurs entrées de résultats (on parlera d'éléments variables).
Enfin, toutes les informations ne rentrant pas dans les deux catégories mentionnées ci-dessus seront à introduire pour chaque résultat (on parlera d'éléments hypervariables).

Voici les constituants de chacune de ces classes :

a. éléments communs
- code de l'étude
- date du début de l'étude
- type d'aliment
- race
- espèce
- état (phase test ou prétest ou posttest)
- voie d'injection
- ~~volume administré~~
- type d'échantillon
- état alimentaire
- véhicule utilisé.

b. éléments variables
- numéro d'animal
- sexe
- âge de l'animal
- période d'acclimatation
- date d'entrée en quarantaine
- durée de la quarantaine
- poids d'organe
- nom du fournisseur
- valeur du poids
- date de mesure du poids.

c. éléments hypervariables
- méthode d'analyse
- date du résultat
- valeur du résultat
- nom du paramètre biologique
- nom du déterminant
- commentaire.

2.2. Déroulement d'une session d'entrée de données

La première étape de la session consistera à proposer à l'utilisateur le choix suivant :

- aller chercher des résultats pour des animaux faisant partie d'une étude (ceci se fera par lancement d'une session de XYBION pour pouvoir faire une extraction selon un déterminant donné).
- aller chercher des résultats pour des animaux en quarantaine (ceci se fera par lancement d'une session de XYBANI).
- effectuer de nouvelles entrées de résultats (ceci de façon tout-à-fait manuelle).

Ensuite, l'utilisateur aura à compléter les données extraites de XYBION / XYBANI pour obtenir des informations complètes concernant une étude; que cela soit au niveau des caractéristiques individuelles des animaux qui en font partie ou bien au niveau des caractéristiques plus globales.

Ces entrées complémentaires se feront par groupe d'éléments communs aux animaux d'une étude (donc effectuées une seule fois pour tous les animaux), puis par groupe d'éléments hypervariables (donc effectuées autant de fois qu'il y a de résultats pour une étude).

Dans le cas d'une entrée manuelle, il est à noter qu'aucune information ne sera présente (récupérable à partir de XYBION / XYBANI).

On peut souligner ici qu'il est important de disposer d'un maximum d'information, celle-ci étant garante d'une utilisation intéressante de la base de données. En effet, plus il y aura d'informations concernant des animaux et les caractéristiques de l'étude, plus on pourra cibler des groupes déterminés pour des études statistiques.
Supposons ainsi qu'une demande de biologie clinique porte sur des animaux avec des caractéristiques multiples et bien déterminées. Si, lors de l'introduction de données d'animaux répondant à ces critères, on n'a pas bien précisé leurs valeurs, on ne pourra pas faire intervenir ces animaux dans l'étude statistique, alors qu'ils auraient dû y participer si l'introduction avait été correctement réalisée.

C'est lors de cette étape d'entrée de données que sera offerte la possibilité de rejeter certaines valeurs pour lesquelles une justification sera demandée. Les rejets et leur justification seront alors conservés dans un fichier (fichier AUDIT).

Une fois que toutes les informations auront été introduites, une étape automatique de mise à jour de la base de données sera réalisée. Cela consistera précisément en un garnissage des fichiers (RMS) constituant la BD. En d'autres termes, une fois que les fichiers intermédiaires (fichiers comprenant les éléments communs, variables et hypervariables) auront été complétés, l'utilisateur responsable (voir § 2.3.) pourra alors lancer le garnissage proprement dit.
A la fin de cette étape, le contenu des trois fichiers intermédiaires sera détruit.
Il est donc clair que la décision d'effectuer cette étape devra être prise seulement lorsuq'on aura la certitude que tous les éléments concernant une étude ont été introduits.

Rem. : ceci ne signifie pas que toutes les données d'une étude devront être introduites en une seule session. L'étape de garnissage sera une étape bien distincte pouvant être lancée au choix de l'utilisateur.

Ce choix de conception nous est apparu le plus favorable au maintien de la bonne cohérence des données introduites. Il serait en effet délicat d'acter, au fur et à mesure, l'entrée d'une partie des informations d'une étude directement sur les fichiers de la BD; comme on l'a dit plus haut, cela nuirait à la qualité de l'information et donc à la bonne exploitation (statistique entre autres) de la BD.

2.3. Responsable de l'introduction des données

Ces tranferts d'information devront se faire par une personne ayant les droits d'accès nécessaires pour utiliser XYBION / XYBANI et ayant à sa disposition toutes les informations nécessaires.

En outre, au niveau pratique, il est plus intéressant qu'une seule personne se charge de ces introductions; celle-ci étant au courant de ce qui a été, ou non, introduit.

3. Exploitation de la base de données

3.1. Type d'information susceptible d'être obtenu à partir de la BD.

La vocation de cette base de données est de pouvoir effectuer des traitements statistiques portant essentiellement sur les résultats dsiponibles.

Pour ce qui est de la biologie clinique, un certain nombre de critères seront utilisés pour sélectionner exclusivement les résultats. Dans ce cas, on ne s'intéressera donc pas à des statistiques sur le nombre d'animaux, de telle espèce, ... etc (bien que cette base de données n'ait pas été créée dans le but d'en faire ce type d'exploitation, il est tout-à-fait envisageable d'y apporter des extensions si nécessaire).

Voici, pour rappel, les critères de sélection donnés :

1. espèce
2. souche (race)
3. fournisseur
4. sexe
5. âge
6. date d'entrée dans l'animalerie
7. à jeûn ou non
8. type d'échantillon
9. nom paramètre biologique
10. méthode d'analyse.

C'est chaque fois à partir de ces dix critères que se feront les demandes statistiques de biologie clinique. Il est entendu qu'il ne faudra pas nécessairement introduire des valeurs pour chacun des ces dix critères. Il faut cependant souligner que plus il y aura de critères spécifiés, plus la recherche d'information sera ciblée et plus celle-ci sera rapide pour l'utilisateur.

En ce qui concerne la toxicologie, on s'intéressera plutôt à des statistiques portant sur l'évolution des résultats repertoriés au cours d'une étude. Par exemple, il est intéressant de connaître la variation de poids pour un animal donné et de la comparer avec l'évolution moyenne des individus.

Rem. : des précisions sur le type de demandes qu'il serait utile de développer peuvent toujours être apportées ainsi que les critères de sélection.

3.2. Type de résultats pouvant être obtenus à partir de cette BD

Des résultats statistiques simples pourront être obtenus. Il est, par exemple, intéressant de pouvoir disposer des renseignements tels que :
- la moyenne de la population
- la médiane
- l'écart-type, la variance
- l'intervalle de confiance
- les valeurs limites.

Ceci, soit pour un même groupe d'animaux, soit pour l'ensemble des animaux (sur une période déterminée ou globalement).

A côté de cela, des traitements plus complexes seront également offerts, tels des distributions normales et régressions.

4. Extensions

Il est prévu de pouvoir ajouter de nouveaux déterminants et de nouveaux paramètres biologiques en plus de ceux déjà considérés (voir liste ci-jointe).

Ceux-ci pourront se rapporter soit à un déterminant existant, soit à un nouveau déterminant qu'on viendra d'introduire.
Cette possibilité sera offerte directement à l'utilisateur (responsable) sous forme d'un menu à l'écran. Cette fonctionnalité sera déjà intégrée dans la première version du logiciel.

Dans une version future de ce logiciel, de nouveaux types de requêtes, formulées sur la base de données et basées sur d'autres critères que ceux évoqués plus haut, pourraient être prévus.

5. Implémentation

Pour le transfert des données dans la base de données, nous utiliserons essentiellement le langage Powerhouse. Il offre une interface à laquelle les utilisateurs sont déjà familiarisés.

Pour l'exploitation de la base de données , nous aurons certainement recours au langage de programmation Pascal. Nous utiliserons pour certains traitements statistiques le logiciel SAS.

Enfin, nous pouvons signaler que le logiciel développé sera réalisé en anglais .

A N N E X E 3

Liste des abréviations des différents types de statistiques

TA : Taux d'un paramètre biologique en fonction de l'âge

TP : Taux d'un paramètre biologique en fonction du poids

TS : Taux d'un paramètre biologique en fonction du sexe

TAL : Taux d'un paramètre biologique en fonction du type d'aliment

TF : Taux d'un paramètre biologique en fonction du fournisseur

TR : Taux d'un paramètre biologique en fonction de la race

ESP : Taux d'un paramètre biologique en fonction de l'espèce

TI : Taux d'un paramètre biologique en fonction du type d'injection

EPE : Evolution du poids en cours d'étude

EPES: Comparaison en fonction du sexe de l'évolution du poids en cours d'étude

EPA : Evolution du poids en fonction de l'âge

EPR : Comparaison en fonction de la race de l'évolution du poids avec l'âge

EPS : Comparaison en fonction du sexe de l'évolution du poids avec l'âge

A N N E X E 4

LISTE DES PARAMETRES

XYBION VERSION 4.1.2
Y-TABLE PARAMETERS DEFINED ON 07-JUL-88

H-MODULE

DETERMINATION/PARAMETERS			UNITS	DECIMALS
1.0 HEMALOG 8/90		SOP NUMBER: ----------		
1.1	PLT	PLATELETS	x10E9/L	0
1.2	WBC	WHITE BLOOD CELLS	x10E9/L	1
1.3	RBC	RED BLOOD CELLS	x10E12/L	2
1.4	HGB	HEMOGLOBIN	g/L	0
1.5	PCV	PACKED CELL VOLUME	1	3
1.6	MCV	MEAN CORPUSCULAR VOLUME	fL	0
1.7	MCH	MEAN CORPUSCULAR HEMOGLOBIN	pg	1
1.8	MCHC	MEAN CORPUSCULAR HEM. CONC.	g/mL	3
2.0 DIFFERENTIAL CELL COUNT		SOP NUMBER: ----------		
2.1	NEUT	NEUTROPHILS	%	0
2.2	EOSN	EOSINOPHILS	%	0
2.3	BASO	BASOPHILS	%	0
2.4	LYMP	LYMPHOCYTES	%	0
2.5	MONO	MONOCYTES	%	0
3.0 RETICULOCYTES COUNT		SOP NUMBER: ----------		
3.1	RETC	RETICULOCYTES COUNT	o/oo	0
4.0 COAGULATION TESTING		SOP NUMBER: ----------		
4.1	PT	PROTHROMBIN TIME	s	1
4.2	APTT	ACTIVATED PART. THROMBO. TIME	s	1
4.3	PPT	PROTHROMBIN-PROCONVERTIN TIME	s	1
5.0 ELECTROPHORESIS/PROTEINS		SOP NUMBER: ----------		
5.1	PROT	PROTEINS	g/L	0
5.2	ALB	ALBUMIN	%	1
5.3	A1G	ALPHA1 GLOBULIN	%	1
5.4	A2G	ALPHA2 GLOBULIN	%	1
5.5	BG	BETA GLOBULIN	%	1
5.6	GG	GAMMA GLOBULIN	%	1
5.7	A/G	ALBUMIN GLOBULIN RATIO	1	2
6.0 ELECTROPHORESIS/LDH		SOP NUMBER: ----------		
6.1	LDHE	LACTATE DEHYDROGENASE	U/L	0
6.2	LDH1	LACTATE DEHYDROGENASE 1	%	1
6.3	LDH2	LACTATE DEHYDROGENASE 2	%	1
6.4	LDH3	LACTATE DEHYDROGENASE 3	%	1
6.5	LDH4	LACTATE DEHYDROGENASE 4	%	1
6.6	LDH5	LACTATE DEHYDROGENASE 5	%	1

XYBION VERSION 4.1.2
Y-TABLE PARAMETERS DEFINED ON 07-JUL-88

S-MODULE

DETERMINATION/PARAMETERS			UNITS	DECIMALS
1.0 COBAS/FARA I		SOP NUMBER: ----------		
1.1	PROT PROTEINS		g/L	0
1.2	TRIG TRIGLYCERIDES		mmol/L	2
1.3	CREA CREATININE		mcmol/L	0
1.4	LDH LACTATE DEHYDROGENASE		U/L	0
1.5	CA CALCIUM		mmol/L	2
1.6	5-ND 5'NUCLEOTIDASE		U/L	1
1.7	ICDH ISOCITRATE DEHYDROGENASE		U/L	0
1.8	SDH SORBITOL DEHYDROGENASE		U/L	1
1.9	ALT ALANINE AMINOTRANSFERASE		U/L	0
1.10	AST ASPARTATE AMINOTRANSFERASE		U/L	0
1.11	ALP ALKALINE PHOSPHATASE		U/L	0
1.12	GGT GAMMA GLUTAMYL TRANSFERASE		U/L	0
1.13	LAP LEUCINE ARYLAMIDASE		U/L	0
1.14	GLDH GLUTAMATE DEHYDROGENASE		U/L	0
1.15	ALB ALBUMINE		g/L	0
1.16	BILI TOTAL BILIRUBIN		mcmol/L	1
2.0 COBAS/FARA II		SOP NUMBER: ----------		
2.1	GLUC GLUCOSE		mmol/L	2
2.2	CHOL CHOLESTEROL		mmol/L	2
2.3	UREA UREA		mmol/L	2
2.4	DBIL DIRECT BILIRUBIN		mcmol/L	1
3.0 BECKMAN E3A		SOP NUMBER: ----------		
3.1	NA SODIUM		mmol/L	0
3.2	K POTASSIUM		mmol/L	1
3.3	CL CHLORIDE		mmol/L	0
4.0 ELECTROPHORESIS/PROTEINS		SOP NUMBER: ----------		
4.1	ALBU ALBUMIN		%	1
4.2	A1G ALPHA1 GLOBULIN		%	1
4.3	A2G ALPHA2 GLOBULIN		%	1
4.4	BG BETA GLOBULIN		%	1
4.5	GG GAMMA GLOBULIN		%	1
4.6	A/G ALBUMIN GLOBULIN RATIO		1	2
5.0 ELECTROPHORESIS/LDH		SOP NUMBER: ----------		
5.1	LDHE LACTATE DEHYDROGENASE		U/L	0
5.2	LDH1 LACTATE DEHYDROGENASE	1	%	1
5.3	LDH2 LACTATE DEHYDROGENASE	2	%	1
5.4	LDH3 LACTATE DEHYDROGENASE	3	%	1
5.5	LDH4 LACTATE DEHYDROGENASE	4	%	1
5.6	LDH5 LACTATE DEHYDROGENASE	5	%	1

XYBION VERSION 4.1.2
Y-TABLE PARAMETERS DEFINED ON 07-JUL-88

U-MODULE

DETERMINATION/PARAMETERS		UNITS	DECIMALS	
1.0 URINARY VOLUME	SOP NUMBER: ----------			
1.1	0-4 VOLUME 0-4H	mL	1	(*)
1.2	4-24 VOLUME 4-24H	mL	1	(*)
1.3	0-24 VOLUME 0-24H	mL	1	(*)
1.4	0-8 VOLUME 0-8H	mL	1	(*)
1.5	0-16 VOLUME 0-16H	mL	1	(*)

(*) Defined with 0 decimals for large animals in R-module table

2.0 URINE OSMOLALITY	SOP NUMBER: ----------			
2.1	OSM1 OSMOLALITY (0-4H Sample)	mmol/L	0	
2.2	OSM2 OSMOLALITY (4-24H ~)	mmol/L	0	
2.3	OSM3 OSMOLALITY (0-24H ~)	mmol/L	0	
2.4	OSMP PLASMATIC OSMOLALITY	mmol/L	0	

3.0 SEMI-QUANTITATIVE URINALYSIS	SOP NUMBER: ----------			
3.1	PH URINARY PH		1	
3.2	UGLU URINARY GLUCOSE	mmol/L	1	
3.3	UPRO URINARY PROTEINS	g/L	1	
3.4	UBLD URINARY OCCULT BLOOD	mg/L	1	
3.5	UKET URINARY KETONE BODIES	mg/L	0	
3.6	UBIL URINARY BILIRUBIN	mcmol/L	1	
3.7	UURO URINARY UROBILINOGEN	EU/L	1	

4.0 MICROSCOPIC URINALYSIS	SOP NUMBER: ----------			
4.1	URBC URINARY RBC	n/mcL	0	
4.2	UWBC URINARY WBC	n/mcL	0	
4.3	UREP URINARY RENAL EPITHELIAL CELLS	n/mcL	0	
4.4	UTEP URINARY TRANSIT,EPITHELIAL CELLS	n/mcL	0	
4.5	USEP URINARY SQUAMOUS EPITH. CELLS	n/mcL	0	
4.6	UCAS URINARY CASTS	n/mcL	0	
4.7	UCRY URINARY CRYSTALS	n/mcL	0	

5.0 QUANTITATIVE COBAS/FARA URINE	SOP NUMBER: ----------			
5.1	UURE UREA	mmol/L	0	
5.2	U.UA URIC ACID	mmol/L	2	
5.3	U.CR CREATININE	mmol/L	1	

6.0 QUANTITATIVE BECKMAN URINE	SOP NUMBER: ----------			
6.1	UNA SODIUM	mmol/L	0	
6.2	UK POTASSIUM	mmol/L	0	
6.3	UCL CHLORIDE	mmol/L	0	

A N N E X E 5

MANUEL UTILISATEUR

1. INTRODUCTION.

Notre logiciel permet de :

1. stocker des valeurs de référence dans une base de données.
2. faire une exploitation statistique de ces données.

Sur le site, il tourne sur le VAX : CPVAX1. L'account sur lequel il se trouve est actuellement [VMMARC.BD]. La liste des fichiers qui sont nécessaires au logiciel est donnée à la fin du manuel. Il faut veiller à ne PAS LES EFFACER !!

2. SESSION UTILISATEUR.

Le premier menu qui apparait est le suivant :

MODE: ACTION:

```
*****************************************
*    BASE  DE  DONNEES  HISTORIQUE      *
*****************************************
```

01 Entrée de données dans la base de données

02 Exploitation de la base de données

03 Ajout d'un déterminant ou d'un paramètre

04 Mise à jour des tables de valeurs permises

En choisissant l'option 1, l'écran représenté ci-dessous apparaît. En choisissant l'option 2, on a l'écran représenté à la page 9. L'écran correspondant à l'option 3 est représenté à la page 12 et pour l'option 4 à la page 13 .

I. L'entrée de données dans la base de données (option 1)

Le choix proposé consiste à

- soit utiliser des données provenant de XYBION
- soit utiliser des données provenant de XYBANI

- soit utiliser des données entrées manuellement
- soit entrer des données de poids corporels
- soit effectuer une mise à jour de la base de données
  (cette option doit normalement être seulement
  accessible au responsable de la B.D.)

L'écran offert est :

```
MODE:  ACTION:

****************************************
*   ENTREE  DES  DONNEES               *
****************************************

01 Utilisation de données provenant de XYBION.

02 Utilisation de données provenant de XYBANI.

03 Entrées manuelles.

04 Entrée de données de poids corporels

05 Mise a jour de la base de données.
```

La description de chacune de ces possibilités est décrite ci-dessous.

1° Utilisation de données de XYBION.

Pour l'entrée de données provenant de XYBION, l'écran proposé est le suivant :

```
MODE:  ACTION:

        ****************************************
        *  ENTREE  DE  DONNEES  DE  XYBION     *
        ****************************************

        01   Conversion des fichiers extraction

        02   Transfert des données PRETEST

        03   Compléments d'information à apporter (PRETEST)

        04   Transfert des données TEST

        05   Compléments d'information à apporter (TEST)

        06   Garnissage de la B.D.
```

La première étape consiste en la conversion des fichiers extractions (option 01). Pour pouvoir effectuer cette opération, il faut avoir ,PREALABLEMENT, réalisé des extractions dans le logiciel XYBION (c'est-à-dire avoir ouvert une session dans XYBION et avoir fait une ou plusieurs extractions selon un déterminant.

La conversion proprement dite s'effectue à partir des fichiers extraits selon un ou plusieurs déterminants pour un ou plusieurs paramètres biologiques.

La seconde étape consiste au transfert des données de Prétest (option 02). Ce sont donc elles qu'il faut compléter en premier lieu. Ce transfert sert à remplir les fichiers qui seront à compléter plus tard (fichiers contenant les éléments communs, variables et hypervariables).

Pour cela, la troisième étape offre la possibilité d'apporter les compléments d'information nécessaires aux données de Prétest (option 03). Cela se réalise avec l'écran de saisie représenté ci-dessous. Il concerne d'abord les éléments communs à une étude :

```
MODE:F   ACTION:F

*****************************************
*  ENTREE  DES  ELEMENTS  COMMUNS       *
*****************************************

    01 Code étude                :    PRE-GARNI
       Date début étude          :
       Espece [RAT]              :
       Race [DEGOUT]             :
       Etat [TEST]               :    PRE-GARNI
       Type aliment              :
       Voie injection [IM]       :
       Type échantillon [PLASMA]:
       Etat alimentaire [AJEUN] :
       Véhicule utilisé [EAU]    :

    99 ELEMENTS VARIABLES
```

L'utilisateur est invité à compléter successivement les champs présentés. Il est obligé de se mettre en mode "F" (Find) pour travailler . Certains des champs peuvent être pré-garnis en fonction des données extraites de XYBION ou bien en fonction des données déjà

stockées dans la base de données pour l'étude considérée. Dans ce cas, la zone est automatiquement passée. De plus, pour la plupart des champs, des valeurs par défaut sont proposées (taper RETURN).

En fin d'écran, l'utilisateur est invité à compléter un deuxième écran (ELEMVARIABLES) vers lequel il est automatiquement orienté. Celui-ci reprend les données variables d'un individu à l'autre. Il est représenté ci-dessous. Ici aussi, certains champs sont déjà garnis avec des valeurs extraites de XYBION .

```
MODE:F     ACTION:F

*****************************************
* ENTREE  DES  ELEMENTS  VARIABLES      *
*****************************************

     01 Numéro d'animal          :
     02 Sexe                     :     PRE-GARNI

        Données de quarantaine
        ----------------------

     03 Code étude de la quarantaine  :
        Numéro d'animal en lot        :
        Durée quarantaine             :          (unité)  :
        Age en début quarantaine      :          (unité)  :

        Date d'entrée en quarantaine  :

        Données complémentaires
        -----------------------

        Période d'acclimatation :          (unité)  :
        Code fournisseur        :
        Volume administre       :          (unité)  :


     99 ELEMENTS HYPERVARIABLES
```

A la fin de l'écran, l'utilisateur peut compléter, via un troisième écran de saisie, les données relatives aux résultats eux-mêmes. Il doit taper l'indicatif 99 pour arriver à l'écran représenté ci-dessous :

```
MODE:E  ACTION:E

****************************************
*  ENTREE DES ELEMENTS HYPERVARIABLES  *
****************************************

      01  Nom du déterminant           :   PRE-GARNI
      02  Nom du paramètre biologique  :   PRE-GARNI
      03  Méthode analyse              :
      04  Jour de l'étude              :   PRE-GARNI
      05  Date du résultat             :   PRE-GARNI
      06  Valeur du résultat           :   PRE-GARNI
                                          (unité) : PRE-GARNI

      07  Commentaire :
```

On constate qu'il ne reste plus à l'utilisateur qu'à compléter deux champs : la méthode d'analyse et le commentaire ; les autres sont déjà garnis automatiquement avec les valeurs extraites de XYBION !

Une fois ces entrées de données terminées, l'utilisateur doit sauver les valeurs qu'il a entrées, avant de sortir de chacun des trois écrans par lesquels il est passé (taper "U" dans la zone action).

Ceci fait, on choisit l'option 6 du menu d'entrées de données de XYBION (cfr.ci-dessus). Elle consiste à réaliser le garnissage de la B.D. pour ces valeurs de Prétest qu'on vient de compléter.

L'étape suivante est alors le transfert des données de la phase TEST cette fois (option 4). Les mêmes étapes que celles décrites pour les données de Prétest sont suivies. C'est-à-dire compléter d'abord les données (option 5) pour ensuite garnir la B.D. avec celles-ci (option 6).

2° Utilisation de données de XYBANI

L'écran proposé est le suivant :

MODE:F ACTION:F

```
****************************************
*  ENTREE  DE  DONNEES  DE  XYBANI     *
****************************************
```

01 Conversion et transfert

02 Compléments d'information à apporter suite àl'extraction

03 Garnissage de la B.D.

Comme pour les données provenant de XYBION, la première étape consiste à convertir les fichiers provenant des extractions du logiciel XYBANI.

Contrairement à XYBION, il n'y a qu'un seul type de données dans XYBANI : les données de quarantaine. C'est pourquoi les deux étapes de conversion et de transfert vers les écrans de saisie s'effectue en une seule fois (option 1).

Ensuite, le complément d'information à fournir suite aux extractions peut s'effectuer comme décrit au point 1° (option 2).

Enfin le garnissage de la B.D. est réalisé par le choix de l'option 3.

3° Utilisation de données entrées manuellement

L'écran proposé est le suivant :

MODE:E ACTION:E

```
********************************
*  ENTREE  MANUELLE            *
********************************
```

01 Entrée de données

02 Garnissage de la B.D.

L'utilisateur est invité (option 1) à entrer ses données via le même type d'écran que ceux utilisés pour l'entrée de données de XYBION et XYBANI. D'abord l'écran permettant la saisie des éléments communs à une étude, ensuite l'entrée des éléments variables d'un individu à l'autre et enfin les données concernant les résultats eux-mêmes.

Il est à noter qu'il n'y a pas de champs pré-garnis ici, contrairement à ce qu'on trouve pour XYBION/XYBANI. Des valeurs par défaut existent néanmoins pour la plupart des champs.

De plus, il faut comme précédamment sauver les données introduites (Taper "U" dans la zone action) avant de sortir de chaque écran !

4° Entrée de données de poids corporels

L'écran est représenté ci-dessous:

```
MODE:E  ACTION:E

*****************************************
*  ENTREE  DE  DONNEE  DE  POIDS        *
*****************************************

01  Etat [TEST] :                    Espece [RAT] :
02  Code étude quarantaine :         Code étude   :
03  Numéro de lot :                  Numéro d'animal :

DATE MESURE          VALEUR    UNITE      NUM_ANIMAL
-------------------------------------------------

:                    :         :          :
:                    :         :          :
:                    :         :          :
:                    :         :          :
:                    :         :          :
```

L'utilisateur est invité à remplir chaque champ et à introduire les différentes valeurs de poids (jusqu'à dix valeur de poids par écran). Avant de sortir de l'écran, il faut absolument sauver les données introduites.

5° La mise à jour de la base de données.

Cette option doit être utilisée avec la plus grande prudence puisqu'elle permet d'accéder directement aux fichiers de la base de données !! Elle doit donc faire l'objet de contrôles de sécurité. Elle sera réservée à un utilisateur privilégié.

Elle a été conçue pour pouvoir modifier en cas de graves problèmes les données de la B.D. directement.

II. L'exploitation de la base de données (option n°2 du menu général).

L'écran offert est représenté ci-dessous :

```
MODE:  ACTION:

       ******************************************
       *  EXPLOITATION    DE    LA    B.D.      *
       ******************************************

       01   Statistiques en biologie clinique

       02   Statistiques en toxicologie
```

Il offre la possibilité d'effectuer des traitements statistiques pour les données de biologie clinique ou pour celles de toxicologie.

A. Statistiques de biologie clinique

L'écran offert est représenté ci-dessous :

```
MODE:  ACTION:

       ******************************************
       * CRITERES DE SELECTION                  *
       ******************************************

       01  Espece [RAT]                  :

       02  Race [ONLAVEUR]               :

       03  Nom fournisseur [MARCHANT]    :

       04  Sexe  [M]                     :
```

```
05  Age animal [=],[6],[SEM]   :(op): (val): (unité):

06  Date entrée                :(op): (val): (unite):

07  Etat alimentaire  [AJEUN]  :

08  Type d'échantillon [PLASMA]:

09  Nom paramètre biologique   :

10  Méthode d'analyse          :

Statistiques:
-------------

  11  Statistiques de biologie clinique
```

L'écran offre la possibilité à l'utilisateur de sélectionner le ou les critères suivant le(s)quel(s) la requète s'effectuera. Les neuf critères sont facultatifs ; seul le critère "Nom param bio" est obligatoire ! De plus, pour l'âge et la date d'entrée dans l'animalerie, il est possible de demander une sélection suivant un des opérateurs : ">", "<", "=" et on peut spécifier les unités correspondant aux valeurs ("SEM","J","MOIS").

Ensuite, il suffit de choisir (option 11 : statistiques) parmi les types de statistiques disponibles, celles dont on veut se servir. La liste des statistiques possibles est donnée sous la forme d'un écran qui est représenté ci-dessous :

```
MODE:  ACTION:

****************************************
*  STATISTIQUES DE BIOLOGIE CLINIQUE   *
****************************************

Type de statistique :

          A choisir parmi :
          -----------------
                              ESPECE
                              AGE
                              POIDS
                              SEXE
                              TYPE_ALIMENT
                              FOURNISSEUR
                              RACE
                              TYPE INJECTION
```

Après cette sélection de critères, il faut sauver les valeurs introduite (taper "U" dans la zone action). Pour chaque choix donné, voivi les traitements statistiques s'y rapportant :

+ ESPECE : taux d'un paramètre en fonction de l'espèce
+ AGE : taux d'un paramètre en fonction de l'âge
+ POIDS : taux d'un paramètre en fonction du poids
+ SEXE : taux d'un paramètre en fonction du sexe
+ TYPE_ALIMENT : taux d'un paramètre en fonction du type d'aliment
+ FOURNISSEUR : taux d'un paramètre en fonction du fournisseur
+ RACE : taux d'un paramètre en fonction de la race
+ TYPE INJECTION : taux d'un paramètre en fonction du type d'injection

Enfin, il faut sortir du logiciel pour aller sur l'account de RS/1 (CPMIC2:$DJA4:[VMMARC]) et copier les fichiers INFORECHERCHE et le fichier résultat de la recherche effectuée ci-dessus (fichier texte ou "QUIZ"). Il est à noter que ces opérations peuvent être effectuées automatiquement si on insère ces commandes directement dans le programme (ceci sera fait lorsque les problèmes d'"Account" seront résolus).

Dans l'account sur CPMIC2, entrer dans RS/1 (taper "rs1") et là, taper "call statistiques" (c'est le module chargé de lancer les types de statistique demandés).

B. Statistiques de toxicologie

L'utilisateur aura le choix entre différents types de statistiques de toxicologie. L'écran représenté ci-dessous en établit la liste :

MODE: ACTION:

```
**************************************
*   STATISTIQUES SUR LES POIDS       *
**************************************
```

1. Evolution du POIDS en cours d'étude.

2. Evolution du POIDS en cours d'étude suivant le sexe.

3. Evolution du POIDS en fonction de l'AGE.

4. Evolution du POIDS en fonction de l'AGE suivant la RACE.

5. Evolution du POIDS en fonction de l'AGE suivant le SEXE.

Les écrans correspondant à une option du menu présenté ci-dessus doivent encore être élaborés.

Ensuite, comme pour la biologie clinique, les statistiques seront réalisées sur RS/1 (elles sont déjà réalisées).

III. L'ajout d'un déterminant ou d'un paramètre (option 3).

Le menu offert est le suivant :

```
MODE:  ACTION:

  ************************************************************
  * MENU GENERAL POUR L'AJOUT D'UN NOUVEAU PARAMETRE OU      *
  * DETERMINANT                                              *
  ************************************************************


    Pour ajouter un nouveau déterminant (D)

    Pour ajouter un nouveau paramètre (P)


   Choix (D/P) :
```

Le premier choix est l'entrée d'un nouveau déterminant. Pour cela, l'utilisateur doit compléter les champs de l'écran suivant :

```
MODE:  ACTION:

************************************************************
*  AJOUT D'UN NOUVEAU DETERMINANT                          *
************************************************************

01  Nom du déterminant   :

02  Nom du paramètre     :

03  Unité de mesure      :
```

Le deuxième choix permet l'entrée d'un nouveau paramètre. L'utilisateur est amené à compléter le même type d'écran que celui représenté ci-dessus.

IIII. La mise à jour des tables de valeurs permises(option4).

Cette dernière option permet la mise à jour des différentes tables de valeurs permises pour certains critères ou paramètres. L'écran suivant est proposé :

```
MODE:  ACTION:

**************************************************************
* MENU GENERAL POUR LA MISE A JOUR DES TABLES DE VALEURS     *
* PERMISES                                                   *
**************************************************************

     01  MAJ de la table VEHICULE UTILISE

     02  MAJ de la table TYPE ECHANTILLON

     03  MAJ de la table VOIE D'INJECTION

     04  MAJ de la table TYPE ALIMENT

     05  MAJ de la table ESPECE

     06  MAJ de la table RACE
```

L'utilisateur a le choix de la table qu'il veut mettre à jour en fonction de nouvelles informations qu'il veut pouvoir traiter !

FICHIERS

. ANIMAL
. CL_CODE
. CL_NUMERO
. CONDEXPERIMENT
. DETERMINANT
. ELEMCOMMUNS
. ELEMVARIABLES
. ELEHYPERVARIABLES
. ESPECE
. ETUDES
. EVOPOIDSAGE
. EVOPOIDSETUDE
. FICH_CONVERTI
. FOURNISSEUR
. FRUIT_PRE_SELECTION_5
. FRUIT_SELECTION_1
. FRUIT_SELECTION_2
. FRUIT_SELECTION_3
. FRUIT_SELECTION_4
. FRUIT_SELECTION_5
. FRUIT_SELECTION_6
. INT_COND_EXP
. INT_COND_EXP_TEMP
. NUMMAX
. PARAMBIOL
. POIDS
. POIDSINTER
. PRETEST
. PRE_INT_COND_EXP
. RACE
. REQUETE_BIOL
. RESULTAT
. SELECTION_DETERMINANT
. STOCKAGE
. TABLE_MESURE
. TAB_CHANGEMENTS
. TEST
. TYPE_ALIMENT
. TYPE_ECHANTILLON
. VEHICULE_UTILISE
. VOIE_INJECTION

CONCEPTION D'UNE BASE DE DONNEES PORTANT SUR DES PARAMETRES BIOLOGIQUES ET ETUDES STATISTIQUES.

Institut d'informatique

Année 1988-1989

Promoteur: M. Noirhomme

Mémoire présenté pour l'obtention du titre de Licencié et Maître en informatique par

François-xavier Fries et
Vincent Marchant.

A N N E X E 6

C O D E D U L O G I C I E L

E C R A N S (QUICK)

MENU_GENERAL

```
screen menu_general menu auto


;-----------------------------------------------------------------
draw thick 4,5 to 8,75
skip to 6
hilite title blinking inverse
title " B A S E   D E   D O N N E E S   H I S T O R I Q U E   " &
centered
;-----------------------------------------------------------------

align (11,14,57)
skip 4
hilite id inverse
subscreen menu_entree_donnee &
      clear all auto &
      label "Entree de donnees dans la base de donnees"

skip 2
subscreen menu_exploitation_bd &
        clear all auto &
        label "Exploitation de la base de donnee"

skip 2
subscreen menu_ajout_general &
        clear all auto mode e &
        label  "Ajout d'un determinant ou d'un parametre"

skip 2
subscreen menu_modif_tables &
        clear all auto &
        label "Mise_a_jour des tables de valeurs permises"

build
```

MENU_ENTREE_DONNEE

```
screen menu_entree_donnee menu


;-------------------------------------------------------------
draw thick 4,5 to 6,75
skip to 5
hilite title inverse
title "    E N T R E E    D E S    D O N N E E S    " &
centered
;-------------------------------------------------------------

align (11,14,57)
skip 4
hilite id inverse
subscreen menu_entrees_xybion &
      clear all auto &
      label "Utilisation de donnees provenant de XYBION"

skip 2
subscreen menu_entrees_xybani &
        clear all auto &
        label "Utilisation de donnees provenant de XYBANI"

skip 2
subscreen menu_entrees_manuelles &
        clear all auto  &
        label  "Entrees manuelles"
skip 2
subscreen ecran_poids_menu &
        clear all auto  &
        label  "Entrée de poids corporels"

skip 2
subscreen menu_maj_bd &
        clear all auto  &
        label  "Mise-a-jour de la base de donnees"

build
```

MENU_ENTREES_XYBION

```
screen menu_entrees_xybion menu

file elemcommuns designer CLOSE
file elemvariables designer CLOSE
file elemvariables alias elemvariables_bis designer CLOSE
file elemvariables alias elemvariables_ter designer CLOSE
file elemhypervariables designer CLOSE
file cl_numero reference
file cl_numero alias cl_numero_bis reference
file pretest reference
file pretest alias pretest_bis reference
file pretest alias pretest_ter reference
file tab_changements reference


;-----------------------------------------------------------------
draw thick 4,5 to 6,75
skip to 5
hilite title  inverse
title " E N T R E E   D E   D O N N E E S   D E   X Y B I O N " &
centered
;-----------------------------------------------------------------

align (11,14,60)
skip 4

hilite title inverse
title "1"
hilite title off
Title "Conversion des fichiers extraction"  at 10,14
skip 1

hilite title inverse
title "2"
hilite title off
Title "Transfert des données PRETEST " at 12,14
skip 1

hilite title inverse
title "3"
hilite title off
Title "Compléments d'information à apporter (PRETEST)" at 14,14
skip 1

hilite title inverse
title "4"
hilite title off
Title "Transfert des données TEST" at 16,14
skip 1

hilite title inverse
title "5"
hilite title off
Title "Compléments d'information à apporter (TEST)" at 18,14
skip 1


hilite title inverse
Title "6"
hilite title off
```

```
title "Garnissage de la B.D." at  20,14


Procedure DESIGNER 1
  begin
    get elemcommuns sequential optional
    if accessok
       then
         error "Vous avez encore des informations à transférer dans la B.D!"
       else
         begin
           clear screen
           do external conv_xybion
           refresh screen
         end
    CLOSE elemcommuns
  end

Procedure DESIGNER 2
  begin
    get elemcommuns sequential optional
    if accessok
       then
         error "Vous avez encore des informations à transférer dans la B.D!"
    CLOSE elemcommuns
    clear screen
    run command "qtp auto=qtp_exe:transfert_pretest.qtc"
    refresh screen
  end

Procedure DESIGNER 3
  begin
    clear screen
    run screen elements_communs_ion mode F
    refresh screen
  end

Procedure DESIGNER 4
  begin
    get elemcommuns sequential optional
    if accessok
       then
         error "Vous avez encore des informations à transférer dans la B.D!"
    CLOSE elemcommuns
    clear screen
    run command "qtp auto=qtp_exe:transfert_test.qtc"
    refresh screen
  end

Procedure DESIGNER 5
  begin
    clear screen
    run screen elements_communs_ion mode F clear all
    refresh screen
  end

Procedure DESIGNER 6
begin
  get elemcommuns sequential
```

```
  if espece of elemcommuns = " "
    then error "Les données pour le garnissage sont incomplètes !"
  else
   BEGIN

; (Vérification de la complétude des données des fichiers elemvariables
;   et elemhypervariables  )

    while retrieving elemvariables sequential
      begin
        if vol_administre of elemvariables = 0
          then
           error "Toutes les caractéristiques des animaux n'ont pas été entrées !"
      end

    while retrieving elemhypervariables sequential
      begin
        if methode_analyse of elemhypervariables = " "
          then
       error "Toutes les caractéristiques des résultats n'ont pas été entrées !"
      end

    CLOSE elemvariables
    CLOSE elemhypervariables

; ( Fin des tests )

    if espece of elemcommuns = "RAT"
    then
     begin
      if etat of elemcommuns = "PRETEST"
        then
          begin
            get elemvariables sequential
            get cl_numero via nr_lot using nr_animal of elemvariables optional
            if accessok
              then
                begin
;(Run command "qtp auto=qtp_exe:garnissage_rat_pre_test.qtc )
                  clear screen
                  Run command "qtp auto=qtp_exe:garnissage_quar.qtc"
                  refresh screen
                end
              else
                begin
; ( Run command "qtp auto=qtp_exe:garnissage_rat_pre.qtc" )
                  clear screen
                  Run command "qtp auto=qtp_exe:garnissage_quar.qtc"
                  refresh screen
                end
          end
        else
          if etat of elemcommuns = "TEST"
            then
              begin
                clear screen
                run command "qtp auto=qtp_exe:garnissage_rat_test.qtc"
                refresh screen
              end
     end
```

```
    else
      if etat of elemcommuns = "PRETEST"
        then
          begin
            get elemvariables_bis sequential
            get pretest via num_animal using (nr_animal of elemvariables + &
                              code_etude_bidon of elemvariables) optional
            if accessok
              then
                begin
                  clear screen
                  run command "qtp auto=qtp_exe:garnissage_nr_pre_quar.qtc"
                  refresh screen
                end
              else
                begin
                  get cl_numero_bis via nr_lot &
                        using nr_animal of elemvariables optional
                  if accessok
                    then
                      begin
                       clear screen
; (run command "qtp auto=qtp_exe:garnissage_nr_pre_test.qtc" )
                       run command "qtp auto=qtp_exe:garnissage_quar.qtc"
                       refresh screen
                      end
                    else
                      begin
                       clear screen
;(  run command "qtp auto=qtp_exe:garnissage_nr_pre.qtc"
;   On lance le garnissage nr_pre_quar au lieu de nr_pre car on ne peut pas
; se baser seulement sur le premier animal pour vérifier l'existence de
; données de quarantaine pour les animaux de l'étude car à une étude peut
; correspondre plusieurs études de quarantaine . Dans ce cas, on pourrait
; avoir des valeurs de quarantaine pour une seule de ces études et donc
; pour un certain nombre seulement d'animaux de l'étude définitive. )

                       run command "qtp auto=qtp_exe:garnissage_nr_pre_quar.qtc"
                       refresh screen
                      end
                end
          end
        else
          if etat of elemcommuns = "TEST"
            then
              begin
                get elemvariables_ter sequential
                get tab_changements via cle_tab using (code_etude_bidon of &
                      elemvariables + code_etude of elemcommuns)
                get pretest_bis via num_animal using (nr_lot &
            of tab_changements + code_etude_bidon of elemvariables) optional
                if accessok
                  then
                    begin
                      clear screen
                      run command "qtp auto=qtp_exe:garnissage_nr_test_quar.qtc"
                      refresh screen
                    end
                  else
                    begin
```

```
                    get pretest_ter via num_animal using &
                     (nr_lot of tab_changements + code_etude of elemcommuns)&
                     optional
                    if accessok
                      then
                        begin
;( "run command qtp auto=qtp_exe:garnissage_nr_test_pre.qtc"
;   il y a une équivalence entre ce que fait le garnissage nr_test_pre
;   et le garnissage rat_test ; d'où l'utilisation du même)
                            clear screen
                            run command &
                                "qtp auto=qtp_exe:garnissage_rat_test.qtc"
                            refresh screen
                          end
                        else
                          begin
                            clear screen
;("qtp auto=qtp_exe:garnissage_nr_test.qtc"
;   On lance le garnissage nr_test_quar au lieu de nr_test car on ne peut pas
; se baser seulement sur le premier animal pour vérifier l'existence de
; données de quarantaine pour les animaux de l'étude car à une étude peut
; correspondre plusieurs études de quarantaine . Dans ce cas, on pourrait
; avoir des valeurs de quarantaine pour une seule de ces études et donc
; pour un certain nombre seulement d'animaux de l'étude définitive. )
                            run command &
                               "qtp auto=qtp_exe:garnissage_nr_test_quar.qtc"
                            refresh screen
                          end
                    end
              end
   END

; ( Effacement de la table tab_changements apr]s un garnissage )

run command "$delete elemcommuns.dat;*"
run command "$delete elemvar.dat;*"
run command "$delete elemhypervar.dat;*"
run command "$copy [.fichvide]elemcommuns.dat;1 [vmmarc.bd]"
run command "$copy [.fichvide]elemvar.dat;1 [vmmarc.bd]"
run command "$copy [.fichvide]elemhypervar.dat;1 [vmmarc.bd]"
run command "$delete tab_changements.dat;*"
run command "$copy [.fichvide]tab_changements.dat;1 [vmmarc.bd]"

end

build
```

```
screen elements_communs_ion
description of screen &
  " Entrée des données communes aux animaux d'une étude de XYBION"

file elemcommuns primary
file animal reference
  access via code_etude using code_etude

file etudes reference
file espece reference
file race reference
file type_aliment reference
file voie_injection reference
file type_echantillon reference
file vehicule_utilise reference

file table_mesure designer

;*************************************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title " E N T R E E   D E S   E L E M E N T S   C O M M U N S " centered
skip to 6
align (3,8,37)
FIELD CODE_ETUDE OF ELEMCOMMUNS display &
              label "Code étude définitif"
FIELD DATE_DEBUT_ETUDE OF ELEMCOMMUNS  id same
skip 2
FIELD ESPECE OF ELEMCOMMUNS  lookup on espece via espece using espece of &
                             elemcommuns &
                             message "Cette espèce est inconnue !" &
                             label "Espece [RAT] "  id same
FIELD RACE OF ELEMCOMMUNS  lookup on race via race using &
                                    race of elemcommuns &
                           message "Cette race est inconnue !" &
                           label "Race [DEGOUT] "id same
FIELD ETAT OF ELEMCOMMUNS  label "Phase de l'étude " display id same
skip 2
FIELD TYPE_ALIMENT OF ELEMCOMMUNS lookup on type_aliment via type_aliment &
                                  using type_aliment of elemcommuns &
                                  message "Ce type d'aliment est inconnu !" &
                                  id same
FIELD VOIE_INJECTION OF ELEMCOMMUNS lookup on voie_injection &
                                    via voie_injection using voie_injection &
                                    of elemcommuns &
                                 message "Cette voie d'injection est inconnue !" &
                                 label "Voie d'injection [IM]" &
                                    id same

FIELD TYPE_ECHANTILLON OF ELEMCOMMUNS lookup on type_echantillon &
                                      via type_echantillon &
                                      using type_echantillon of elemcommuns &
                                   message "Ce type d'échantillon est inconnu !" &
                                      label "Type échantillon [PLASMA] " id same
FIELD ETAT_ALIMENTAIRE OF ELEMCOMMUNS label "Etat alimentaire [AJEUN] " id same
FIELD VEHICULE_UTILISE OF ELEMCOMMUNS  lookup on vehicule_utilise &
                                       via vehicule_utilise using &
                                       vehicule_utilise of elemcommuns &
                                message "Ce véhicule utilisé est inconnu !" &
```

```
                                    label "Véhicule utilisé [EAU]" &
                                        id same
skip 1
SUBSCREEN elements_variables_ion passing elemcommuns &
          auto mode same label "ELEMENTS VARIABLES" refresh all id 99



Procedure ENTRY
begin
  error "On ne peut compléter des données en MODE ENTRY dans cet écran !"
end

Procedure DESIGNER 1
  begin
    get animal via code_etude using code_etude of elemcommuns optional
    if accessok
      then
        begin
          get etudes via code_etude using code_etude of elemcommuns
          let date_debut_etude = date_debut_etude of etudes
          display date_debut_etude
          let race = race of animal
          display race
          let espece = espece of animal
          display espece
          information "Cette étude a déjà été introduite dans la B.D. !"
        end
    if date_debut_etude of elemcommuns <= 0
      then ACCEPT date_debut_etude
    if espece of elemcommuns <= " "
      then ACCEPT espece
    if race of elemcommuns <= " "
      then ACCEPT race
    if etat of elemcommuns = "TEST"
      then ACCEPT type_aliment
    ACCEPT voie_injection of elemcommuns
    ACCEPT type_echantillon of elemcommuns
    ACCEPT etat_alimentaire of elemcommuns
    ACCEPT vehicule_utilise of elemcommuns
  end

Procedure INPUT date_debut_etude
  begin
    if matchpattern(fieldtext,"!0")
     then
       if date_debut_etude LE 0
         then error "You MUST enter a value"
  end


Procedure INPUT espece
  begin
    if matchpattern(fieldtext,"!0")
     then let fieldtext = "RAT"
  end

Procedure INPUT race
  begin
    if matchpattern(fieldtext,"!0")
```

```
      then let fieldtext = "DEGOUT"
  end

Procedure PROCESS etat
begin
  if etat = "QUAR"
    then
      error "Ce menu ne permet pas l'introduction de données de quarantaine"
end

Procedure INPUT voie_injection
  begin
    if matchpattern(fieldtext,"!0")
         then let fieldtext = "INTRAMUSCULAIRE"
  end


Procedure INPUT type_echantillon
  begin
    if matchpattern(fieldtext,"!0")
     then let fieldtext = "PLASMA"
  end

Procedure INPUT etat_alimentaire
  begin
    if matchpattern(fieldtext,"!0")
     then let fieldtext = "AJEUN"
  end

Procedure INPUT vehicule_utilise
  begin
    if matchpattern(fieldtext,"!0")
         then let fieldtext = "EAU"
  end


Procedure EXIT
begin
  if etat of elemcommuns = "TEST"
     then
        begin
          clear screen
          run COMMAND "QTP auto=qtp_exe:maj_numanimal_test.qtc"
        end
     else
       if etat of elemcommuns = "PRETEST"
         then
           begin
             clear screen
             run COMMAND "QTP auto=qtp_exe:maj_numanimal_pretest.qtc"
           end
end

build
```

```
screen elements_variables_ion receiving elemcommuns

description of screen &
    " Entrée des éléments variables d'un animal à l'autre de XYBION"

file elemcommuns master need 1
file elemvariables primary

file tab_changements designer
file fournisseur designer
file num_max designer
file table_mesure designer
file elemhypervariables designer close

define d_date_debut_etude date = date_debut_etude of elemcommuns
define cle_animal char*16 = nr_animal of elemvariables &
                                + code_etude of elemcommuns
define cle_animal_quar char*16 = nr_lot of tab_changements + &
                        code_etude_bidon of elemvariables
define cle_nr_lot_code_def char*16 = nr_lot of tab_changements + &
                        code_etude of elemcommuns
define cle_nr_def_code_bidon char*16 = nr_animal of elemvariables &
                        + code_etude_bidon of elemvariables

define d_nr_lot char*8 = nr_lot of tab_changements
define d_nr_def char*8 = nr_def of tab_changements
define d_code_bidon char*8 = code_bidon of tab_changements
define d_code_def char*8 = code_def of tab_changements


temporary t_date_res date
temporary unite_per_acclimatation char*8 initial "j"
temporary unite_age_animal char*8 initial "j"
temporary unite_duree_quarantaine char*8 initial "j"
temporary nouveau_fournisseur char*1 initial "V"
temporary unite_vol_administre char*8 initial "ml"

temporary bool_quar char*1 initial "F"
temporary bool_etude char*1 initial "F"
temporary tab_changements_put char*1 initial "V"

file pretest reference
file pretest alias pretest_bis reference
file pretest alias pretest_ter reference
file pretest alias pretest_qua reference
file pretest alias pretest_cin reference
file cl_numero reference

file animal reference
file animal alias animal_bis reference
file animal alias animal_ter reference
file animal alias animal_qua reference
file animal alias animal_cin reference

;*****************************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "E N T R E E   D E S   E L E M E N T S   V A R I A B L E S " centered
skip to 6
```

```
align (3,8,38)
;*************************************************************************

FIELD NR_ANIMAL OF ELEMVARIABLES
FIELD SEXE OF ELEMVARIABLES display
skip 1
hilite title underline
title "Données de quarantaine :"  at 9,3
align(3,8,38)
FIELD CODE_ETUDE_BIDON of elemvariables   &
                    label "Code étude de la quarantaine" upshift
FIELD NR_LOT OF ELEMVARIABLES  &
    label "Numéro d'animal en lot" id same
FIELD DUREE OF ELEMVARIABLES  label "Durée quarantaine" id same
align (,,50)
field unite_duree_quarantaine predisplay &
     values "j","an","mois","sem" id same

align (3,8,38)
FIELD AGE_ANIMAL OF ELEMVARIABLES &
              label "Age en début quarantaine" id same
align (,,50)
field unite_age_animal predisplay &
     values "j","an","mois","sem" id same
align (3,8,38)

FIELD DATE_ENTREE OF ELEMVARIABLES &
                label "Date entrée en quarantaine" id same
skip 1
title "Données complémentaires:" at 16,3
FIELD PER_ACCLIMATATION OF ELEMVARIABLES id same
align (,,50)
field unite_per_acclimatation predisplay &
     values "j","an","mois","sem" id same
align (3,8,38)
FIELD NOM_FOURNISSEUR OF ELEMVARIABLES id same &
              label "Code fournisseur"
FIELD vol_administre OF ELEMVARIABLES id same
align(,,54)
field unite_vol_administre predisplay  &
                        values "ml","cl","dl","l","micl" id same

align  (3,8,38)
skip 1

subscreen elements_hypervariables_ion passing elemvariables,&
                    d_nr_lot,d_nr_def,d_code_bidon,d_code_def,&
                    d_date_debut_etude, tab_changements_put &
          auto label "ELEMENTS HYPERVARIABLES" mode same &
          refresh all id 99


Procedure ENTRY
begin
    error &
  "Il faut se mettre en MODE FIND pour compléter des données dans cet écran !!"
end

Procedure DESIGNER 1
begin
```

```
  if etat of elemcommuns = "PRETEST"
    then ACCEPT nr_animal of elemvariables
    else WARNING "Ce numéro ne peut être changé en phase TEST !"
end

Procedure PROCESS nr_lot
begin
  let nr_lot of tab_changements = nr_lot of elemvariables
  get cl_numero via nr_def using nr_animal optional
  if accessok
    then
      if nr_lot of cl_numero <> nr_lot of tab_changements
        then
          error "Ce numéro de lot ne correspond pas à ce numéro d'étude !"
end

Procedure DESIGNER 3
begin
    if espece of elemcommuns <> "RAT"
      then
        begin
          ACCEPT code_etude_bidon of elemvariables
        end
    if etat of elemcommuns = "TEST"
      then
        begin
          ACCEPT nr_lot of elemvariables
        end

;(Recherche de données éventuellement déjà présentes dans la B.D. pour les
; afficher si nécessaire : on effectue cette opération ici car on est sur que
; l'utilisateur passe par ici )

    if etat of elemcommuns = "PRETEST"
     then
        begin
          let nr_lot of tab_changements = nr_animal
          if espece of elemcommuns = "RAT"
            then
              begin
                get pretest via num_animal using cle_animal optional
                if accessok
                  then
                    begin
                      let duree = duree of pretest
                      display duree
                      let age_animal = age_animal of pretest
                      display age_animal
                      let per_acclimatation = per_acclimatation of pretest
                      display per_acclimatation
                      let date_entree = date_entree of pretest
                      display date_entree
                      let bool_etude = "V"
                      let bool_quar = "V"
                    end
              end
            else
              begin
                get pretest via num_animal using cle_animal optional
                if accessok
```

```
            then
              begin
                let duree = duree of pretest
                display duree
                let age_animal = age_animal of pretest
                display age_animal
                let per_acclimatation = per_acclimatation of pretest
                display per_acclimatation
                let date_entree = date_entree of pretest
                display date_entree
                let bool_etude = "V"
                let bool_quar = "V"
              end
            else
              begin
                get pretest_bis via num_animal &
                    using cle_nr_def_code_bidon optional
                if accessok
                  then
                    begin
                      let age_animal = age_animal of pretest_bis
                      display age_animal
                      let date_entree = date_entree of pretest_bis
                      display date_entree
                      let bool_quar = "V"
                    end
              end
         end
       if espece of elemcommuns <> "RAT"
         then
           begin
             get tab_changements via cle_tab &
                using code_etude_bidon of elemvariables + &
                        code_etude of elemcommuns optional
             if not accessok
               then
                 begin
                   let code_bidon of tab_changements = code_etude_bidon of &
                              elemvariables
                   let code_def of tab_changements = code_etude &
                                                      of elemcommuns
                   let nr_def of tab_changements = " "
                   let nr_lot of tab_changements = " "
                 end
               else
                 let tab_changements_put = "F"
            end
   end
else
  if etat of elemcommuns= "TEST"
    then
      begin
        if espece of elemcommuns = "RAT"
          then
            begin
              get pretest_ter via num_animal &
                  using cle_nr_lot_code_def optional
              if accessok
                then
                  begin
```

```
                    let duree = duree of pretest_ter
                    display duree
                    let age_animal = age_animal of pretest_ter
                    display age_animal
                    let per_acclimatation = per_acclimatation of pretest_ter
                    display per_acclimatation
                    let date_entree = date_entree of pretest_ter
                    display date_entree
                    let bool_etude = "V"
                    let bool_quar = "V"
                   end

; (MISE_A_JOUR DE TAB_CHANGEMENTS)

                    let code_def of tab_changements = code_etude of elemcommuns
                    let nr_def of tab_changements = nr_animal of elemvariables
                    let code_bidon of tab_changements = " "
                  end
            else
              begin
                get pretest_qua via num_animal &
                      using cle_nr_lot_code_def optional
                if accessok
                  then
                    begin
                      let duree = duree of pretest_qua
                      display duree
                      let age_animal = age_animal of pretest_qua
                      display age_animal
                      let per_acclimatation = per_acclimatation of pretest_qua
                      display per_acclimatation
                      let date_entree = date_entree of pretest_qua
                      display date_entree
                      let bool_etude = "V"
                      let bool_quar = "V"
                    end
                  else
                    begin
                      get pretest_cin via num_animal using cle_animal_quar &
                          optional
                      if accessok
                        then
                          begin
                            let age_animal = age_animal of pretest_cin
                            display age_animal
                            let date_entree = date_entree of pretest_cin
                            display date_entree
                            let bool_quar = "V"
                          end
                    end

; (MISE_A_JOUR DE TAB_CHANGEMENTS)

                  let code_bidon of tab_changements = code_etude_bidon &
                                                  of elemvariables
                  let code_def of tab_changements = code_etude of elemcommuns
                  let nr_def of tab_changements = nr_animal of elemvariables
                end
          end
```

```
;(Recherche du fournisseur de l'animal pour l'afficher si on le connait déjà)

   if espece of elemcommuns = "RAT"
      then
        begin
           if etat of elemcommuns = "PRETEST"
             then
               begin
                  get animal via num_animal using cle_animal optional
                  if accessok
                    then
                       begin
                         LET SEXE OF ELEMVARIABLES = SEXE OF ANIMAL
                         get fournisseur via num_fournisseur &
                                        using num_fournisseur of animal
                          let nom_fournisseur of elemvariables = nom_fournisseur &
                                                                 of fournisseur
                          display nom_fournisseur
                          let num_fournisseur of elemvariables = num_fournisseur &
                                                                 of fournisseur
                          end
             end
             else
                if etat of elemcommuns = "TEST"
                   then
                     begin
                       get animal_bis via num_animal &
                         using cle_nr_lot_code_def optional
                       if accessok
                          then
                            begin
                              LET SEXE OF ELEMVARIABLES = SEXE OF ANIMAL_bis
                              get fournisseur via num_fournisseur &
                                        using num_fournisseur of animal_bis
                              let nom_fournisseur of elemvariables = &
                                        nom_fournisseur of fournisseur
                              display nom_fournisseur
                              let num_fournisseur of elemvariables = &
                                        num_fournisseur of fournisseur
                           end
                     end
        end
      else
        if espece of elemcommuns <> "RAT"
           then
             begin
               if etat of elemcommuns = "PRETEST"
                 then
                   begin
                     get animal via num_animal using cle_animal optional
                     if accessok
                       then
                         begin
                           LET SEXE OF ELEMVARIABLES = SEXE OF ANIMAL
                           get fournisseur via num_fournisseur &
                                        using num_fournisseur of animal
                           let nom_fournisseur of elemvariables = &
                                        nom_fournisseur  of fournisseur
                           display nom_fournisseur
```

```
            let num_fournisseur of elemvariables = &
                        num_fournisseur of fournisseur
          end
        else
          begin
            get animal_ter via num_animal &
               using cle_nr_def_code_bidon  optional
            if accessok
              then
                begin
                  LET SEXE OF ELEMVARIABLES = SEXE OF ANIMAL_TER
                  get fournisseur via num_fournisseur &
                        using num_fournisseur of animal_ter
                  let nom_fournisseur of elemvariables = &
                        nom_fournisseur  of fournisseur
                  display nom_fournisseur
                  let num_fournisseur of elemvariables = &
                           num_fournisseur of fournisseur
                end
          end
end
  else
    if etat of elemcommuns = "TEST"
        then
           begin
              get animal_qua via num_animal &
                 using cle_nr_lot_code_def optional
              if accessok
                then
                   begin
                   LET SEXE OF ELEMVARIABLES = &
                                    SEXE OF ANIMAL_qua
                     get fournisseur via num_fournisseur &
                         using num_fournisseur of animal_qua
                     let nom_fournisseur of elemvariables = &
                          nom_fournisseur of fournisseur
                     display nom_fournisseur
                     let num_fournisseur of elemvariables = &
                          num_fournisseur of fournisseur
                   end
                else
                   begin
                     get animal_cin via num_animal &
                         using cle_animal_quar optional
                     if accessok
                       then
                         begin
                           LET SEXE OF ELEMVARIABLES = &
                               SEXE OF ANIMAL_cin
                          get fournisseur via num_fournisseur &
                             using num_fournisseur &
                                 of animal_cin
                         let nom_fournisseur of elemvariables &
                            = nom_fournisseur of fournisseur
                         display nom_fournisseur
                         let num_fournisseur of elemvariables &
                            = num_fournisseur of fournisseur
                           end
                   end
              end
```

```
            end
  if bool_etude = "F"
    then
      begin
        ACCEPT duree
        ACCEPT unite_duree_quarantaine
      end
  if bool_quar = "F"
    then
      begin
         ACCEPT age_animal
         ACCEPT unite_age_animal
         ACCEPT date_entree
      end

;(contrôle que la date_entree en quarantaine est bien inférieure à la date
; du début de l'étude car on est en phase pretest ou test )

  if date_entree of elemvariables >= date_debut_etude of elemcommuns
    then error &
 "La date de début étude est erronnée par rapport à la date_entrée_quarantaine!"

; (contrôle que la date d'entrée n'est pas supérieure à la date du résultat
; de elemhypervariables)

  get elemhypervariables via num_animal using num_animal of elemvariables
  while retrieving elemhypervariables
    begin
      if etat of elemcommuns = "PRETEST"
        then
          begin
            let t_date_res = date(days(d_date_debut_etude)+jour_etude &
                             of elemhypervariables)
            if t_date_res >= d_date_debut_etude
              then
                begin
                  error &
 "La date des résultats est supérieure à la date du début de l'étude!"
                  break
                end
            if t_date_res < date_entree of elemvariables
              then
                begin
                  error &
 "La date des résultats est inférieure à la date du début de quarantaine!"
                  break
                end
          end
        else
         if etat of elemcommuns = "TEST"
           then
            begin
             let t_date_res = date(days(d_date_debut_etude)+(jour_etude &
                              of elemhypervariables - 1))
             if t_date_res < d_date_debut_etude
              then
               begin
                  error &
"La date des résultats est inférieure à la date du début de l'étude!"
                  break
```

```
            end
          end
    end

  if bool_etude = "F"
    then
      begin
        ACCEPT per_acclimatation
        ACCEPT unite_per_acclimatation
      end
  if nom_fournisseur <= " "
    then ACCEPT nom_fournisseur
  ACCEPT vol_administre
  ACCEPT unite_vol_administre

end

Procedure INPUT duree
begin
   if matchpattern(fieldtext,"!0")
     then
       if duree LE 0
         then
            error "You MUST enter a value"
end

procedure PROCESS unite_duree_quarantaine
  begin
    if unite_duree_quarantaine <> "j"
       then
         begin
           get table_mesure via entree using unite_duree_quarantaine
           while retrieving table_mesure
             begin
               if sortie = "j"
                 then
                   begin
                     let unite_duree_quarantaine = "j"
                     let duree = (duree * facteur of table_mesure)
                     display duree
                     break
                   end
             end
        end
  end


Procedure INPUT age_animal
begin
   if matchpattern(fieldtext,"!0")
     then
       if age_animal LE 0
         then
            error "You MUST enter a value"
end

procedure PROCESS unite_age_animal
  begin
    if unite_age_animal <> "j"
       then
```

```
        begin
          get table_mesure via entree using unite_age_animal
          while retrieving table_mesure
            begin
              if sortie = "j"
                then
                  begin
                    let unite_age_animal = "j"
                    let age_animal = (age_animal * facteur of table_mesure)
                    display age_animal
                    break
                  end
            end
        end
  end


Procedure INPUT per_acclimatation
begin
   if matchpattern(fieldtext,"!0")
     then
       if per_acclimatation LE 0
         then
            error "You MUST enter a value"
end


procedure PROCESS unite_per_acclimatation
  begin
    if unite_per_acclimatation <> "j"
       then
         begin
           get table_mesure via entree using unite_per_acclimatation
           while retrieving table_mesure
             begin
               if sortie = "j"
                 then
                   begin
                     let unite_per_acclimatation = "j"
                     let per_acclimatation = (per_acclimatation * facteur &
                                              of table_mesure)
                     display per_acclimatation
                     break
                   end
             end
        end
  end


Procedure INPUT date_entree
begin
   if matchpattern(fieldtext,"!0")
     then
       if date_entree LE 0
         then
            error "You MUST enter a value"
end


Procedure INPUT nom_fournisseur
```

```
begin
   if matchpattern(fieldtext,"!0")
     then
       if nom_fournisseur LE " "
         then
            error "You MUST enter a value"
end

procedure PROCESS nom_fournisseur
  begin
    get fournisseur via nom_fournisseur using nom_fournisseur of elemvariables &
              optional
    if accessok
       then
         begin
           let num_fournisseur = num_fournisseur of fournisseur
         end
       else
         begin
           get num_max sequential
           let num_fournisseur of num_max = num_fournisseur of num_max + 1
           let num_fournisseur = num_fournisseur of num_max
           let num_fournisseur of fournisseur = num_fournisseur of num_max
           put num_max
           let nom_fournisseur of fournisseur = nom_fournisseur of elemvariables
           put fournisseur
         end
  end


Procedure INPUT vol_administre
begin
  if matchpattern(fieldtext,"!0")
    then
       if vol_administre LE 0
         then
            error "You MUST enter a value"
end

procedure PROCESS unite_vol_administre
  begin
    if unite_vol_administre <> "ml"
       then
         begin
           get table_mesure via entree using unite_vol_administre
           while retrieving table_mesure
             begin
               if sortie = "ml"
                 then
                   begin
                     let unite_vol_administre = "ml"
                     let vol_administre = (vol_administre * facteur &
                                          of table_mesure)
                     display vol_administre
                     break
                   end
             end
        end
  end

build
```

```
screen elements_hypervariables_ion receiving elemvariables,&
                        d_nr_lot,d_nr_def,d_code_bidon,d_code_def, &
                                          d_date_debut_etude, &
                                          tab_changements_put
description of screen &
    "Entrée des éléments variables d'un résultat à l'autre pour un même animal"

file elemvariables master need 1
file elemhypervariables primary

file parambiol reference
file elemcommuns designer
file tab_changements designer
file elemhypervariables alias elemhypervariables_bis designer

define d_nr_lot char*8
define d_nr_def char*8
define d_code_def char*8
define d_code_bidon char*8
define d_date_debut_etude date

temporary t_resultat date
temporary tab_changements_put char*1
temporary unite_valeur_res char*8
temporary existant char*4
temporary var_num_animal char*16

;*************************************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "E N T R E E   D E S   E L E M E N T S   H Y P E R V A R I A B L E S" &
   centered
skip to 6
align (3,8,38)
;*************************************************************************

FIELD NOM_DET OF ELEMHYPERVARIABLES display label "Nom du déterminant"
FIELD NOM_PARAM_BIO OF ELEMHYPERVARIABLES display

FIELD METHODE_ANALYSE OF ELEMHYPERVARIABLES
FIELD JOUR_ETUDE OF ELEMHYPERVARIABLES display label "Jour de l'étude"
FIELD DATE_RES OF ELEMHYPERVARIABLES display nochange
FIELD VALEUR_RES OF ELEMHYPERVARIABLES display
align (,,52)
field unite_valeur_res display
align (3,8,20)
skip 1
;align (,,1)
FIELD COMMENTAIRE OF ELEMHYPERVARIABLES id at 13,3 &
        label "Commentaire :" at 13,8 data at 14,1

Procedure INPUT methode_analyse
  begin
    if matchpattern(fieldtext,"!0")
      then
        if methode_analyse LE " "
           then error "You MUST enter a value"
  end
```

```
Procedure PROCESS methode_analyse
  begin
    get parambiol via nom_param_bio using nom_param_bio of elemhypervariables
    if accessok
       then
         begin
           let unite_valeur_res = unite_mesure of parambiol
           display unite_valeur_res
         end
       else
         error "Le parametre n'a pas encore été repertorié  dans la B.D. "
    get elemcommuns sequential
    if etat of elemcommuns = "TEST"
      then
        let t_resultat = date(days(d_date_debut_etude)+jour_etude - 1)
      else
        let t_resultat = date(days(d_date_debut_etude)+jour_etude)

    let date_res = t_resultat
    display date_res
  end

Procedure ENTRY
begin
  error "Se mettre en mode FIND pour compléter les données !"
end

Procedure  PREUPDATE
  begin
    if not deletedrecord
      then
        begin

; ( Test pour voir si ces valeurs n'ont pas encore été introduites)

          let var_num_animal = num_animal of elemhypervariables
          get elemhypervariables_bis via num_animal using var_num_animal &
                                    optional
          if accessok
            then
              while retrieving elemhypervariables_bis
                begin
                  if nom_det of elemhypervariables_bis = nom_det of &
                                 elemhypervariables
                    then
                      if nom_param_bio of elemhypervariables_bis = &
                              nom_param_bio of elemhypervariables
                        then
                          if methode_analyse of elemhypervariables_bis = &
                             methode_analyse of elemhypervariables
                             then
                               if jour_etude of elemhypervariables_bis = &
                                  jour_etude of elemhypervariables
                                  then
                                    if valeur_res of elemhypervariables_bis = &
                                       valeur_res of elemhypervariables
                                       then
                     error "Vous avez déjà introduit ces valeurs pour cet animal"

                  end
```

```
; ( Fin test)

          if tab_changements_put = "V"
            then
              begin
                get tab_changements via cle_tab using (d_code_bidon + &
                          d_code_def) optional
                if accessok
                  then
                    begin
                      while retrieving tab_changements
                        begin
                          if d_nr_lot <> nr_lot of tab_changements
                            then
                              let existant = "FAUX"
                            else
                              if d_nr_def <> nr_def of tab_changements
                                then
                                   let existant = "FAUX"
                                else
                                   begin
                                     let existant = "VRAI"
                                     break
                                   end
                        end
                      if existant = "FAUX"
                         then
                           begin
                             let nr_lot of tab_changements = d_nr_lot
                             let nr_def of tab_changements = d_nr_def
                             let code_bidon of tab_changements = d_code_bidon
                             let code_def of tab_changements = d_code_def
                             put tab_changements
                           end
                    end
                  else
                    begin
                      let nr_lot of tab_changements = d_nr_lot
                      let nr_def of tab_changements = d_nr_def
                      let code_bidon of tab_changements = d_code_bidon
                      let code_def of tab_changements = d_code_def
                      put tab_changements
                    end
              end
        end
  end

BUILD
```

```
screen menu_entrees_xybani menu

;file determinant reference
;file cl_numero reference
;file cl_code reference
file elemcommuns designer CLOSE
file elemvariables designer CLOSE
file elemhypervariables designer CLOSE

;-------------------------------------------------------------
draw thick 4,5 to 6,75
skip to 5
hilite title  inverse
title " E N T R E E   D E   D O N N E E S   D E   X Y B A N I  " &
centered
;-------------------------------------------------------------

align (11,15,60)
skip 5

hilite title inverse
title "01"
hilite title off
title "Conversion et transfert " at 11,15

skip 2
hilite id inverse
subscreen elements_communs_ani mode F &
      clear all &
      label "Compléments d'information à apporter suite à l'extraction" &
      id 2

hilite title inverse
title "03" at 17,11
hilite title off
title "Garnissage de la B.D." at 17,15


Procedure DESIGNER 1
  begin
    get elemcommuns sequential optional
    if accessok
       then
         error "Vous avez encore des informations à transférer dans la B.D.!"
       else
         begin
           clear screen
           do external conv_xybion
           refresh screen
           CLOSE elemcommuns
;( On utilise ces fichiers dans TRANSFERT ! Il faut donc les fermer ici! )
           clear screen
           run command "qtp auto=qtp_exe:transfert.qtc"
           refresh screen
         end
  end

Procedure DESIGNER 3
begin
  get elemcommuns sequential
```

```
  if espece of elemcommuns = " "
     then
       error "Les données pour le garnissage sont incomplètes !"
     else
      BEGIN
; (Vérification de la complétude des données des fichiers elemvariables et &
;  elemhypervariables )

       while retrieving elemvariables sequential
         begin
           if vol_administre of elemvariables = 0
             then
        error "Toutes les caractéristiques des animaux n'ont pas été entrées !"
         end

       while retrieving elemhypervariables sequential
         begin
           if methode_analyse of elemhypervariables = " "
             then
      error "Toutes les caractéristiques des résultats n'ont pas été entrées !"
         end

; ( Fin des tests)

       CLOSE elemcommuns
       CLOSE elemvariables
       CLOSE elemhypervariables
;( On utilise ces fichiers dans GARNISSAGE ! Il faut donc les fermer ici! )
       clear screen
       run command "qtp auto=qtp_exe:garnissage_quar.qtc"
       refresh screen
;(On ne gère que ce garnissage ici!)

       run command "$delete elemcommuns.dat;*"
       run command "$copy [vmmarc.fichvide]elemcommuns.dat;1 [vmmarc.bd]"
       run command "$delete elemvar.dat;*"
       run command "$copy [vmmarc.fichvide]elemvar.dat;1 [vmmarc.bd]"
       run command "$delete elemhypervar.dat;*"
       run command "$copy [vmmarc.fichvide]elemhypervar.dat;1 [vmmarc.bd]"
;(Effacage de TAB_CHENGEMENTS ou sont consignés les correspondances de
; numéros pour une introduction de données)

      END

;  get cl_code via code_bidon using code_etude of elemcommuns optional
;  if accessok
;    then
;      begin
;        get elemvariables sequential
;        get cl_numero via nr_lot using nr_animal of elemvariables optional
;        if accessok
;          then
;            begin
;              clear screen
;              run command "qtp auto=qtp_exe:garnissage_quar_test.qtc"
;              refresh screen
;            end
;          else
;            begin
;              clear screen
```

```
;               run command "qtp auto=qtp_exe:garnissage_quar_pretest.qtc"
;               refresh screen
;             end
;       end
;     else
;       begin
;         clear screen
;         run command "qtp auto=qtp_exe:garnissage_quar.qtc"
;         refresh screen
;       end
end

build
```

```
screen elements_communs_ani
description of screen &
  " Entrée des données communes aux animaux d'une étude de XYBANI"

file elemcommuns primary
file animal reference
  access via code_etude using code_etude

file etudes reference
file espece reference
file race reference
file voie_injection reference
file type_echantillon reference
file vehicule_utilise reference

file table_mesure designer


;*************************************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title " E N T R E E   D E S   E L E M E N T S   C O M M U N S " centered
skip to 6
align (3,8,44)
FIELD CODE_ETUDE OF ELEMCOMMUNS display &
             label "Code étude attribué en quarantaine"
FIELD DATE_DEBUT_ETUDE OF ELEMCOMMUNS  id same
skip 2
FIELD ESPECE OF ELEMCOMMUNS  lookup on espece via espece using espece of &
                             elemcommuns &
                             message "Cette espèce est inconnue !" &
                             label "Espece [SINGE] "  id same
FIELD RACE OF ELEMCOMMUNS  lookup on race via race using race of elemcommuns &
                           message "Cette race est inconnue !" &
                           label "Race [ALUNETTE] "  id same
FIELD ETAT OF ELEMCOMMUNS  label "Phase de l'étude " display id same
skip 2


FIELD VOIE_INJECTION OF ELEMCOMMUNS lookup on voie_injection &
                                    via voie_injection using voie_injection &
                                    of elemcommuns &
                               message "Cette voie d'injection est inconnue !" &
                                    label "Voie_injection [IM]" id same


FIELD TYPE_ECHANTILLON OF ELEMCOMMUNS lookup on type_echantillon &
                                      via type_echantillon &
                                      using type_echantillon of elemcommuns &
                                 message "Ce type d'échantillon est inconnu !" &
                                      label "Type échantillon [PLASMA] " id same
FIELD ETAT_ALIMENTAIRE OF ELEMCOMMUNS label "Etat alimentaire [AJEUN] " id same
FIELD VEHICULE_UTILISE OF ELEMCOMMUNS  lookup on vehicule_utilise &
                                       via vehicule_utilise using &
                                       vehicule_utilise of elemcommuns &
                                  message "Ce véhicule utilisé est inconnu !" &
                                       label "Vehicule utilise [EAU]" id same
skip 1
SUBSCREEN elements_variables_ani passing elemcommuns &
```

```
          auto mode same label "ELEMENTS VARIABLES" refresh all id 99

Procedure INPUT date_debut_etude
  begin
    if matchpattern(fieldtext,"!0")
     then
       if date_debut_etude LE 0
         then error "You MUST enter a value"
  end

Procedure PROCESS date_debut_etude
  begin
    get animal via code_etude using code_etude of elemcommuns optional
    if accessok
      then
        begin
          get etudes via code_etude using code_etude of elemcommuns
          let date_debut_etude = date_debut_etude of etudes
          let race = race of animal
          display race
          let espece = espece of animal
          display espece
          information "Cette étude a déjà été introduite dans la B.D. !"
        end
  end

Procedure INPUT espece
  begin
    if matchpattern(fieldtext,"!0")
     then let fieldtext = "SINGE"
     else
        if matchpattern(fieldtext,"RAT")
          then
             error "Il ne peut exister des données de rongeurs dans XYBANI !"
  end

Procedure INPUT race
  begin
    if matchpattern(fieldtext,"!0")
      then let fieldtext = "ALUNETTE"
    let etat = "QUAR"
  end



Procedure INPUT voie_injection
  begin
    if matchpattern(fieldtext,"!0")
         then let fieldtext = "INTRAMUSCULAIRE"
  end


Procedure INPUT type_echantillon
  begin
    if matchpattern(fieldtext,"!0")
     then let fieldtext = "PLASMA"
  end

Procedure INPUT etat_alimentaire
  begin
```

```
    if matchpattern(fieldtext,"!0")
     then let fieldtext = "AJEUN"
  end

Procedure INPUT vehicule_utilise
  begin
    if matchpattern(fieldtext,"!0")
         then let fieldtext = "EAU"
  end

Procedure ENTRY
begin
  error "Se mettre en mode FIND pour compléter les données !"
end

build
```

```
screen elements_variables_ani receiving elemcommuns

description of screen &
    " Entrée des éléments variables d'un animal à l'autre de XYBANI"

file elemcommuns master need 1
file elemvariables primary

file fournisseur designer
file num_max designer
file table_mesure designer
file elemhypervariables designer close

file pretest reference
file pretest alias pretest_bis reference
file animal reference
file animal alias animal_bis reference
file cl_code reference

define d_date_debut_etude date = date_debut_etude of elemcommuns
define cle_animal char*16 = nr_animal of elemvariables &
                                  + code_etude of elemcommuns
temporary t_date_res date
temporary unite_age_animal char*8 initial "j"
temporary unite_duree_quarantaine char*8 initial "j"
temporary nouveau_fournisseur char*1 initial "V"
temporary unite_vol_administre char*8 initial "ml"
temporary bool_quar char*1 initial "F"



;*************************************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "E N T R E E   D E S   E L E M E N T S   V A R I A B L E S " centered
skip to 6
align (3,8,38)
;*************************************************************************

FIELD NR_ANIMAL OF ELEMVARIABLES display &
    label "Numéro d'animal en lot"
FIELD SEXE OF ELEMVARIABLES display
skip 1

hilite title underline
title "Données de quarantaine :"  at 9,3
;FIELD DUREE OF ELEMVARIABLES required  label "Durée quarantaine" id at ,3
;align (,,50)
;field unite_duree_quarantaine predisplay &
;     values "j","an","mois","sem" id same

align (6,8,38)
FIELD AGE_ANIMAL OF ELEMVARIABLES if bool_quar = "F" &
         label "Age en début quarantaine"  id same
align (,,50)
field unite_age_animal predisplay &
     values "j","an","mois","sem" id same
align (6,8,38)
```

```
;FIELD PER_ACCLIMATATION OF ELEMVARIABLES id same
;align (,,50)
;field unite_per_acclimatation predisplay &
;     values "j","an","mois","sem" id same
;align (6,11,38)

FIELD DATE_ENTREE OF ELEMVARIABLES if bool_quar = "F" &
            label "Date entrée en quarantaine" id same

skip 1
title "Données fournisseur:" at 14,3
align (6,8,38)
FIELD NOM_FOURNISSEUR OF ELEMVARIABLES if nom_fournisseur <= " " id same &
              label "Code fournisseur"
align (3,8,38)

skip 2
FIELD vol_administre OF ELEMVARIABLES id same
align(,,54)
field unite_vol_administre predisplay  &
                        values "ml","cl","dl","l","micl" id same

align  (3,8,38)
skip 1

subscreen elements_hypervariables_ani passing elemvariables, &
          d_date_debut_etude &
          auto label "ELEMENTS HYPERVARIABLES" mode same &
          refresh all id 99


Procedure designer 2
begin
  get cl_code via code_bidon using code_etude of elemcommuns optional
  if accessok
    then
      begin
        get animal via num_animal using (nr_animal of elemvariables + code_def &
                    of cl_code)
        get fournisseur via num_fournisseur using num_fournisseur of animal
        let nom_fournisseur of elemvariables = nom_fournisseur of fournisseur
        let num_fournisseur of elemvariables = num_fournisseur of fournisseur
        get pretest via num_animal using (nr_animal of elemvariables + &
                        code_def of cl_code)
        let age_animal = age_animal of pretest
        let date_entree = date_entree of pretest
        let bool_quar = "V"
      end
   else
      begin
        get animal_bis via num_animal using cle_animal optional
        if accessok
          then
            begin
              get fournisseur via num_fournisseur using num_fournisseur &
                          of animal_bis
              let nom_fournisseur of elemvariables = nom_fournisseur of &
                          fournisseur
              let num_fournisseur of elemvariables = num_fournisseur of &
                          fournisseur
```

```
            end
        get pretest_bis via num_animal using cle_animal optional
        if accessok
          then
            begin
              let age_animal = age_animal of pretest_bis
              let date_entree = date_entree of pretest_bis
              let bool_quar = "V"
            end
     end
; TEST DEMANDE
   if bool_quar = "F"
      then
        begin
          accept age_animal
          accept unite_age_animal
          accept date_entree
        end
      else
        begin
          display age_animal
          display date_entree
        end

;(contrôle que la date_entree en quarantaine est bien égale à la date du
; début de l'étude)

  if date_entree of elemvariables <> date_debut_etude of elemcommuns
    then error &
 "La date de début étude est erronnée par rapport à la date_entrée_quarantaine!"

; (contrôle que la date d'entrée n'est pas supérieure à la date du résultat
; de elemhypervariables)

  get elemhypervariables via num_animal using num_animal of elemvariables
  while retrieving elemhypervariables
    begin
      let t_date_res = date(days(d_date_debut_etude)+(jour_etude &
                       of elemhypervariables - 1))
      if t_date_res < date_debut_etude of elemcommuns
       then
         begin
           error "La date d'entrée est supérieure à la date du résultat!"
           break
         end
    end

   if nom_fournisseur = " "
      then
         accept nom_fournisseur
      else
         display nom_fournisseur
   accept vol_administre
   accept unite_vol_administre

; FIN TEST
end

Procedure INPUT age_animal
begin
```

```
   if matchpattern(fieldtext,"!0")
     then
       if age_animal LE 0
         then
            error "You MUST enter a value"
end

procedure PROCESS unite_age_animal
  begin
    if unite_age_animal <> "j"
       then
         begin
           get table_mesure via entree using unite_age_animal
           while retrieving table_mesure
             begin
               if sortie = "j"
                 then
                   begin
                     let unite_age_animal = "j"
                     let age_animal = (age_animal * facteur of table_mesure)
                     display age_animal
                     break
                   end
             end
        end
  end



Procedure INPUT date_entree
begin
   if matchpattern(fieldtext,"!0")
     then
       if date_entree LE 0
         then
            error "You MUST enter a value"
end


Procedure INPUT nom_fournisseur
begin
   if matchpattern(fieldtext,"!0")
     then
       if nom_fournisseur LE " "
         then
            error "You MUST enter a value"
end

procedure PROCESS nom_fournisseur
  begin
    get fournisseur via nom_fournisseur using nom_fournisseur of elemvariables &
              optional
    if accessok
       then
         begin
           let num_fournisseur = num_fournisseur of fournisseur
         end
       else
         begin
           get num_max sequential
```

```
          let num_fournisseur of num_max = num_fournisseur of num_max + 1
          let num_fournisseur = num_fournisseur of num_max
          let num_fournisseur of fournisseur = num_fournisseur of num_max
          put num_max
          let nom_fournisseur of fournisseur = nom_fournisseur of elemvariables
          put fournisseur
        end
  end



Procedure INPUT vol_administre
begin
  if matchpattern(fieldtext,"!0")
    then
       if vol_administre LE 0
         then
            error "You MUST enter a value"
end

procedure PROCESS unite_vol_administre
  begin
    if unite_vol_administre <> "ml"
       then
         begin
           get table_mesure via entree using unite_vol_administre
           while retrieving table_mesure
             begin
               if sortie = "ml"
                 then
                   begin
                     let unite_vol_administre = "ml"
                     let vol_administre = (vol_administre * facteur &
                                          of table_mesure)
                     display vol_administre
                     break
                   end
             end
       end
  end

Procedure ENTRY
begin
  error "Se mettre en mode FIND pour compléter les données !"
end

BUILD
```

```
screen elements_hypervariables_ani receiving elemvariables, d_date_debut_etude
description of screen &
    "Entrée des éléments variables d'un résultat à l'autre pour un même animal"

file elemvariables master need 1
file elemhypervariables primary
file parambiol reference
file elemhypervariables alias elemhypervariables_bis designer

define d_date_debut_etude date
define d_resultat date = date(days(d_date_debut_etude)+ (jour_etude - 1))

temporary var_num_animal char*16
temporary unite_valeur_res char*8

;*************************************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "E N T R E E   D E S   E L E M E N T S   H Y P E R V A R I A B L E S" &
   centered
skip to 6
align (3,8,38)
;*************************************************************************

FIELD NOM_DET OF ELEMHYPERVARIABLES display label "Nom du déterminant"
FIELD NOM_PARAM_BIO OF ELEMHYPERVARIABLES display

FIELD METHODE_ANALYSE OF ELEMHYPERVARIABLES
FIELD JOUR_ETUDE OF ELEMHYPERVARIABLES display label "Jour de l'étude"
FIELD DATE_RES OF ELEMHYPERVARIABLES display nochange
FIELD VALEUR_RES OF ELEMHYPERVARIABLES display
align (,,52)
field unite_valeur_res display
align (3,8,20)
skip 1
;align (,,1)
FIELD COMMENTAIRE OF ELEMHYPERVARIABLES id at 13,3 &
        label "Commentaire :" at 13,8 data at 14,1

Procedure INPUT methode_analyse
  begin
    if matchpattern(fieldtext,"!0")
      then
        if methode_analyse LE " "
           then error "You MUST enter a value"
  end

Procedure PROCESS methode_analyse
  begin
    get parambiol via nom_param_bio using nom_param_bio of elemhypervariables
    if accessok
       then
         begin
           let unite_valeur_res = unite_mesure of parambiol
           display unite_valeur_res
         end
       else
         error "Le parametre n'a pas encore été repertorié  dans la B.D. "
    let date_res = d_resultat
```

```
    display date_res
  end


Procedure ENTRY
begin
  error "Se mettre en mode FIND pour compléter les données !"
end

Procedure  PREUPDATE
  begin
    if not deletedrecord
      then
        begin

; ( Test pour voir si ces valeurs n'ont pas encore été introduites)

          let var_num_animal = num_animal of elemhypervariables
          get elemhypervariables_bis via num_animal using var_num_animal &
                                    optional
          if accessok
            then
              while retrieving elemhypervariables_bis
                begin
                  if nom_det of elemhypervariables_bis = nom_det of &
                                elemhypervariables
                    then
                      if nom_param_bio of elemhypervariables_bis = &
                              nom_param_bio of elemhypervariables
                        then
                          if methode_analyse of elemhypervariables_bis = &
                             methode_analyse of elemhypervariables
                             then
                               if jour_etude of elemhypervariables_bis = &
                                 jour_etude of elemhypervariables
                                 then
                                   if valeur_res of elemhypervariables_bis = &
                                      valeur_res of elemhypervariables
                                      then
                     error "Vous avez déjà introduit ces valeurs pour cet animal"

                end

; ( Fin test)

        end
  end


build
```

```
screen menu_entrees_manuelles menu

file elemcommuns designer CLOSE
file elemvariables designer CLOSE
file elemhypervariables designer CLOSE
file elemvariables alias elemvariables_bis designer CLOSE
file elemvariables alias elemvariables_ter designer CLOSE
file elemvariables alias elemvariables_qua designer CLOSE
;file cl_code reference
file cl_numero reference
file cl_numero alias cl_numero_bis reference
;file cl_numero alias cl_numero_ter reference
file pretest reference
file pretest alias pretest_bis reference
file pretest alias pretest_ter reference
file tab_changements reference

;-----------------------------------------------------------------
draw thick 4,5 to 6,75
skip to 5
hilite title  inverse
title " E N T R E E   M A N U E L L E     " &
centered
;-----------------------------------------------------------------

align (11,14,57)
skip 4
hilite id inverse
subscreen elements_communs_man &
      clear all &
      label "Entree de donnees "

skip 3

hilite title inverse
Title "02"
hilite title off
title "Garnissage de la B.D." at  14,14



Procedure DESIGNER 2
begin
  get elemcommuns sequential
  if espece of elemcommuns = " "
    then error "Les données pour le garnissage sont incomplètes !"
  else
   BEGIN

; ( Vérification de la complétude des données des fichiers elemvariables
;   et elemhypervariables  )

      While retrieving elemvariables sequential
        begin
          if vol_administre of elemvariables = 0
            then
         error "Toutes les caractéristiques des animaux n'ont pas été entrées !"
        end

      While retrieving elemhypervariables sequential
```

```
        begin
          if methode_analyse of elemhypervariables = " "
            then
        error "Toutes les caractéristiques des résultats n'ont pas été entrées !"
        end

; (Fin des tests )
  CLOSE elemcommuns
  CLOSE elemvariables
  CLOSE elemhypervariables

  if espece of elemcommuns = "RAT"
    then
     begin
      if etat of elemcommuns = "PRETEST"
        then
          begin
            get elemvariables sequential
            get cl_numero via nr_lot using nr_animal of elemvariables optional
            if accessok
              then
                begin
; (Run command "qtp auto=qtp_exe:garnissage_rat_pre_test.qtc")
                  clear screen
                  Run command "qtp auto=qtp_exe:garnissage_quar.qtc"
                  refresh screen
                end
              else
                begin
; (Run command "qtp auto=qtp_exe:garnissage_rat_pre_test.qtc")
                  clear screen
                  Run command "qtp auto=qtp_exe:garnissage_quar.qtc"
                  refresh screen
                end
          end
        else
          if etat of elemcommuns = "TEST"
            then
              begin
                run command "qtp auto=qtp_exe:garnissage_rat_test.qtc"
                clear screen
              end
      end
    else
      if etat of elemcommuns = "PRETEST"
        then
          begin
            get elemvariables_bis sequential
            get pretest via num_animal using (nr_animal of elemvariables + &
                          code_etude_bidon of elemvariables) optional
            if accessok
              then
                begin
                  clear screen
                  run command "qtp auto=qtp_exe:garnissage_nr_pre_quar.qtc"
                  refresh screen
                end
              else
                begin
                  get cl_numero_bis via nr_lot &
```

```
                    using nr_animal of elemvariables optional
                  if accessok
                    then
                      begin
; (Run command "qtp auto=qtp_exe:garnissage_nr_pre_test.qtc")
                        clear screen
                        run command "qtp auto=qtp_exe:garnissage_quar.qtc"
                        refresh screen
                      end
                    else
                      begin
; (Run command "qtp auto=qtp_exe:garnissage_nr_pre.qtc"
; On lance le garnissage nr_pre_quar au lieu de nr_pre car on ne peut pas se
; baser seulement sur le premier animal pour vérifier l'existence de données
; de quarantaine pour les animaux de l'étude car à une étude peut correspondre
; plusieurs études de quarantaine. Dans ce cas, on pourrait avoir des valeurs
; de quarantaine pour une seule de ces études et donc pour un certain nombre
; seulement d'aniamux de l'étude définitive.)
                        clear screen
                        run command &
                           "qtp auto=qtp_exe:garnissage_nr_pre_quar.qtc"
                        refresh screen
                      end
                end
          end
        else
          if etat of elemcommuns = "TEST"
            then
              begin
                get elemvariables_ter sequential
                get tab_changements via cle_tab using (code_etude_bidon of &
                      elemvariables + code_etude of elemcommuns)optional
                get pretest_bis via num_animal using (nr_lot of tab_changements
+ &
                      code_etude_bidon of elemvariables)optional
                if accessok
                  then
                    begin
                      clear screen
                      run command "qtp auto=qtp_exe:garnissage_nr_test_quar.qtc"
                      refresh screen
                    end
                  else
                    begin
                      get pretest_ter via num_animal using &
                       (nr_lot of tab_changements + code_etude of elemcommuns)&
                            optional
                      if accessok
                        then
                          begin
; (Run command "qtp auto=qtp_exe:garnissage_nr_test_pre.qtc"
; il y a une équivalence entre ce que fait le garnissage nr_test_pre et
; le garnissage rat_test ; d'où l'utilisation du même )
                            clear screen
                            run command &
                              "qtp auto=qtp_exe:garnissage_rat_test.qtc"
                            refresh screen
                          end
                        else
                          begin
; (Run command "qtp auto=qtp_exe:garnissage_nr_test.qtc"
```

```
; On lance le garnissage nr_test_quar au lieu de nr_test car on ne peut pas se
; baser seulement sur le premier animal pour vérifier l'existence de données
; de quarantaine pour les animaux de l'étude car à une étude peut correspondre
; plusieurs études de quarantaine. Dans ce cas, on pourrait avoir des valeurs
; de quarantaine pour une seule de ces études et donc pour un certain nombre
; seulement d'aniamux de l'étude définitive.)
                              clear screen
                              run command &
                                "qtp auto=qtp_exe:garnissage_nr_test_quar.qtc"
                              refresh screen
                            end
                  end
            end
          else
            if etat of elemcommuns = "QUAR"
              then
                begin
                  clear screen
                  run command "qtp auto=qtp_exe:garnissage_quar.qtc"
                  refresh screen

;                 get cl_code via code_bidon &
;                             using code_etude of elemcommuns optional
;                 if accessok
;                   then
;                     begin
;                       get elemvariables_qua sequential
;                       get cl_numero_ter via nr_lot using nr_animal &
;                             of elemvariables optional
;                       if accessok
;                        then
;                          begin
;                            clear screen
;                            run command &
;                              "qtp auto=qtp_exe:garnissage_quar_test.qtc"
;                            refresh screen
;                          end
;                        else
;                          begin
;                            clear screen
;                            run command &
;                              "qtp auto=qtp_exe:garnissage_quar_pretest.qtc"
;                            refresh screen
;                          end
;                     end
;                   else
;                     begin
;                       clear screen
;                       run command "qtp auto=qtp_exe:garnissage_quar.qtc"
;                       refresh screen
;                     end
;                 end
   END

; (On efface le fichier tab_changements apr]s un garnissage )


run command "$delete elemcommuns.dat;*"
run command "$delete elemvar.dat;*"
run command "$delete elemhypervar.dat;*"
```

```
run command "$copy [.fichvide]elemcommuns.dat;1 [vmmarc.bd]"
run command "$copy [.fichvide]elemvar.dat;1 [vmmarc.bd]"
run command "$copy [.fichvide]elemhypervar.dat;1 [vmmarc.bd]"

run command "$delete tab_changements.dat;*"
run command "$copy [.fichvide]tab_changements.dat;1 [vmmarc.bd]"

end

build
```

```
screen elements_communs_man
description of screen &
  " Entrée des données communes aux animaux d'une étude"

file elemcommuns primary
file animal reference
  access via code_etude using code_etude

file elemcommuns alias elemcommuns_bis designer

file etudes reference
file espece reference
file race reference
file type_aliment reference
file voie_injection reference
file type_echantillon reference
file vehicule_utilise reference

file table_mesure designer

temporary t_code_etude_entre char*8

;*************************************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title " E N T R E E   D E S   E L E M E N T S   C O M M U N S " centered
skip to 6
align (3,8,37)
FIELD CODE_ETUDE OF ELEMCOMMUNS required &
              label "Code étude définitif"
FIELD DATE_DEBUT_ETUDE OF ELEMCOMMUNS if date_debut_etude <= 0 required
skip 2
FIELD ESPECE OF ELEMCOMMUNS  if espece <= " " &
                             lookup on espece via espece using espece of &
                             elemcommuns &
                             message "Cette espèce est inconnue !" &
                             label "Espece [RAT] "
FIELD RACE OF ELEMCOMMUNS  if race <= " " &
                           lookup on race via race using race of elemcommuns &
                           message "Cette race est inconnue !" required &
                            label "Race [DEGOUT]"

FIELD ETAT OF ELEMCOMMUNS  label "Phase de l'étude [TEST] " required
skip 2
FIELD TYPE_ALIMENT OF ELEMCOMMUNS lookup on type_aliment via type_aliment &
                                  using type_aliment of elemcommuns &
                                  message "Ce type d'aliment est inconnu !" &
                                  if etat of elemcommuns = "TEST"
FIELD VOIE_INJECTION OF ELEMCOMMUNS lookup on voie_injection &
                                    via voie_injection using voie_injection &
                                    of elemcommuns &
                                 message "Cette voie d'injection est inconnue !" &
                                       label "Voie d'injection [IM]" required
FIELD TYPE_ECHANTILLON OF ELEMCOMMUNS lookup on type_echantillon &
                                      via type_echantillon &
                                      using type_echantillon of elemcommuns &
                                   message "Ce type d'échantillon est inconnu !" &
                                      label "Type échantillon [PLASMA] "
FIELD ETAT_ALIMENTAIRE OF ELEMCOMMUNS label "Etat alimentaire [AJEUN] "
```

```
FIELD VEHICULE_UTILISE OF ELEMCOMMUNS  lookup on vehicule_utilise &
                                       via vehicule_utilise using &
                                       vehicule_utilise of elemcommuns &
                                  message "Ce véhicule utilisé est inconnu !" &
                                       label "Véhicule utilisé [EAU]" required
skip 1
SUBSCREEN elements_variables_man passing elemcommuns &
          auto mode same label "ELEMENTS VARIABLES" refresh all id 99



Procedure INPUT date_debut_etude
  begin
    if matchpattern(fieldtext,"!0")
     then
       if date_debut_etude LE 0
         then error "You MUST enter a value"
  end

Procedure PROCESS code_etude
  begin
    let t_code_etude_entre = code_etude of elemcommuns
    get elemcommuns_bis sequential optional
    if accessok
       then
         begin
           if t_code_etude_entre = code_etude of elemcommuns_bis
              then
                error "Cette étude a déjà été enregistrée dans ce menu!"
              else
                error "Le garnissage pour l'étude en cours n'a pas été effectué"
         end
    let code_etude of elemcommuns = t_code_etude_entre

    get animal via code_etude using code_etude of elemcommuns optional
    if accessok
      then
        begin
          get etudes via code_etude using code_etude of elemcommuns
          let date_debut_etude = date_debut_etude of etudes
          let race = race of animal
          let espece = espece of animal
          information "Cette étude a déjà été introduite dans la B.D. !"
        end
  end

Procedure INPUT espece
  begin
    if matchpattern(fieldtext,"!0")
     then let fieldtext = "RAT"
  end

Procedure INPUT race
begin
  if matchpattern (fieldtext,"!0")
    then  let fieldtext = "DEGOUT"
end
```

```
Procedure INPUT etat
begin
  if matchpattern(fieldtext,"!0")
    then
      let fieldtext = "TEST"
end


Procedure PROCESS etat
begin
  if etat = "QUAR"
    then
      if espece = "RAT"
        then
          error "Pas de données rat en quarantaine!"
end


Procedure INPUT Voie_injection
begin
  if matchpattern(fieldtext,"!0")
    then let fieldtext = "INTRAMUSCULAIRE"
end


Procedure INPUT type_echantillon
  begin
    if matchpattern(fieldtext,"!0")
     then let fieldtext = "PLASMA"
  end

Procedure INPUT etat_alimentaire
  begin
    if matchpattern(fieldtext,"!0")
     then let fieldtext = "AJEUN"
  end

Procedure INPUT vehicule_utilise
begin
  if matchpattern(fieldtext,"!0")
    then let fieldtext = "EAU"
end

Procedure EXIT
begin
  if etat of elemcommuns = "TEST"
    then
        begin
          clear screen
          information "La mise-à-jour des clés de fichiers se fait....."
          run COMMAND "QTP auto=qtp_exe:maj_numanimal_test.qtc"
        end
end

build
```

```
screen elements_variables_man receiving elemcommuns

description of screen &
    " Entrée des éléments variables d'un animal à l'autre "

file elemcommuns master need 1
file elemvariables primary

file fournisseur designer
file num_max designer
file table_mesure designer
file tab_changements designer

file cl_numero reference
file cl_code reference
file animal reference
file animal alias animal_bis reference
file animal alias animal_ter reference
file animal alias animal_qua reference
file animal alias animal_cin reference
file animal alias animal_six reference
file animal alias animal_seven reference

file pretest reference
file pretest alias pretest_bis reference
file pretest alias pretest_ter reference
file pretest alias pretest_qua reference
file pretest alias pretest_cin reference
file pretest alias pretest_six reference
file pretest alias pretest_seven reference

;temporary champ_erreur integer
;temporary champ_erreur_2 char*30

define d_date_debut_etude date = date_debut_etude of elemcommuns
define cle_animal char*16 = nr_animal of elemvariables &
                                + code_etude of elemcommuns
define cle_animal_quar char*16 = nr_lot of tab_changements + code_etude_bidon &
                                  of elemvariables
define cle_nr_lot_code_def char*16 =  nr_lot of tab_changements + &
                                code_etude of elemcommuns
define cle_nr_def_code_bidon char*16 = nr_animal of elemvariables + &
                                  code_etude_bidon of elemvariables

define d_nr_lot char*8 = nr_lot of tab_changements
define d_nr_def char*8 = nr_def of tab_changements
define d_code_bidon char*8 = code_bidon of tab_changements
define d_code_def char*8 = code_def of tab_changements

temporary unite_per_acclimatation char*8 initial "j"
temporary unite_age_animal char*8 initial "j"
temporary unite_duree_quarantaine char*8 initial "j"
temporary nouveau_fournisseur char*1 initial "V"
temporary unite_vol_administre char*8 initial "ml"

temporary bool_quar char*1 initial "F"
temporary bool_etude char*1 initial "F"
temporary tab_changements_put char*1 initial "V"
```

```
;*********************************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "E N T R E E   D E S   E L E M E N T S   V A R I A B L E S " centered
skip to 6
align (3,8,38)
;*********************************************************************

FIELD NR_ANIMAL OF ELEMVARIABLES required
FIELD NUM_ANIMAL OF ELEMVARIABLES silent nochange
skip 1

;field champ_erreur nolabel noid display
;align(,,50)
;field champ_erreur_2 id same display
;align(3,8,38)

hilite title underline
title "Données de quarantaine :"  at 8,3
align(3,8,38)
FIELD CODE_ETUDE_BIDON OF ELEMVARIABLES if espece of elemcommuns <> "RAT" &
                      and  etat of elemcommuns <> "QUAR" &
                      required label "Code étude de la quarantaine" upshift
FIELD NR_LOT OF ELEMVARIABLES if etat of elemcommuns = "TEST" &
    required label "Numéro d'animal en lot"
FIELD SEXE OF ELEMVARIABLES required
FIELD DUREE OF ELEMVARIABLES if bool_etude = "F"  and &
                              etat of elemcommuns <> "QUAR"&
                         label "Durée quarantaine"
align (,,50)
field unite_duree_quarantaine  if bool_etude = "F"  and &
                              etat of elemcommuns <> "QUAR" predisplay &
     values "j","an","mois","sem" id same

align (3,8,38)

FIELD AGE_ANIMAL OF ELEMVARIABLES if bool_quar = "F" &
                  label "Age en début quarantaine"
align (,,50)
field unite_age_animal if bool_quar = "F" predisplay &
     values "j","an","mois","sem" id same
align (3,8,38)

FIELD DATE_ENTREE OF ELEMVARIABLES if bool_quar = "F" &
                      label "Date entrée en quarantaine"
skip 1
title "Données complémentaires:" at 16,3
FIELD PER_ACCLIMATATION OF ELEMVARIABLES if bool_etude = "F" &
                                    and  etat of elemcommuns <> "QUAR"


align (,,50)
field unite_per_acclimatation if bool_etude = "F"  and &
                              etat of elemcommuns <> "QUAR" predisplay &
     values "j","an","mois","sem" id same
align (3,8,38)
FIELD NOM_FOURNISSEUR OF ELEMVARIABLES  if nom_fournisseur <= " " &
```

```
               label "Code fournisseur"
FIELD vol_administre OF ELEMVARIABLES required
align(,,54)
field unite_vol_administre predisplay  &
                        values "ml","cl","dl","l","micl" id same

align  (3,8,38)
skip 1

subscreen elements_hypervariables_man passing elemvariables,&
                      d_nr_lot ,d_nr_def,&
                      d_code_bidon,&
                      d_code_def ,&
                      d_date_debut_etude, tab_changements_put &
          auto label "ELEMENTS HYPERVARIABLES" mode same &
          refresh all id 99


Procedure PROCESS nr_lot
begin
  let nr_lot of tab_changements = nr_lot of elemvariables
  get cl_numero via nr_def using nr_animal optional
  if accessok
    then
      if nr_lot of cl_numero <> nr_lot of tab_changements
        then
          error "Ce numéro de lot ne correspond pas à ce numéro d'étude !"
end

Procedure INPUT duree
begin
   if matchpattern(fieldtext,"!0")
     then
       if duree LE 0
         then
           error "You MUST enter a value"
end

procedure PROCESS sexe
  begin
   let num_animal of elemvariables = cle_animal

;( Recherche de données éventuellement déjà présentes dans la BD pour les
;  les afficher si nécessaire: on effectue cette opération ici car on est sur
;  que l'utilisateur passe par ici  mode ENTRY )

    if etat of elemcommuns = "PRETEST"
     then
        begin
          let nr_lot of tab_changements = nr_animal
          if espece of elemcommuns = "RAT"
            then
              begin
                get pretest via num_animal using cle_animal optional
                if accessok
                  then
                    begin
                      let duree = duree of pretest
                      let age_animal = age_animal of pretest
                      let per_acclimatation = per_acclimatation of pretest
```

```
                let date_entree = date_entree of pretest
                let bool_etude = "V"
                let bool_quar = "V"
              end
        end
      else
        begin
          get pretest via num_animal using cle_animal optional
          if accessok
            then
              begin
                let duree = duree of pretest
                let age_animal = age_animal of pretest
                let per_acclimatation = per_acclimatation of pretest
                let date_entree = date_entree of pretest
                let bool_etude = "V"
                let bool_quar = "V"
              end
            else
              begin
              get pretest_bis via num_animal using cle_nr_def_code_bidon &
                          optional
                if accessok
                  then
                    begin
                      let age_animal = age_animal of pretest_bis
                      let date_entree = date_entree of pretest_bis
                      let bool_quar = "V"
                    end
              end
        end
      if espece of elemcommuns <> "RAT"
        then
          begin
            get tab_changements via cle_tab &
               using code_etude_bidon of elemvariables + &
                     code_etude of elemcommuns optional
            if not accessok
              then
                begin
                  let code_bidon of tab_changements = code_etude_bidon &
                                                 of elemvariables
                  let code_def of tab_changements = code_etude &
                                                    of elemcommuns
                  let nr_def of tab_changements = " "
                  let nr_lot of tab_changements = " "
                end
              else
                let tab_changements_put = "F"
          end
  end
else
  if etat of elemcommuns= "TEST"
    then
      begin
        if espece of elemcommuns = "RAT"
          then
            begin
              get pretest_ter via num_animal &
                  using cle_nr_lot_code_def  optional
```

```
                if accessok
                  then
                    begin
                     let duree of elemvariables = duree of pretest_ter
                     let age_animal of elemvariables = &
                           age_animal of pretest_ter
                     let per_acclimatation of elemvariables = &
                          per_acclimatation of pretest_ter
                     let date_entree of elemvariables = &
                          date_entree of pretest_ter
                     let bool_etude = "V"
                     let bool_quar = "V"
                    end

; (MISE_A_JOUR DE TAB_CHANGEMENTS)

                     let code_def of tab_changements = code_etude of elemcommuns
                     let nr_def of tab_changements = nr_animal of elemvariables
                     let code_bidon of tab_changements = " "
                  end
             else
               begin
                 get pretest_qua via num_animal &
                       using cle_nr_lot_code_def optional
                 if accessok
                   then
                     begin
                       let duree = duree of pretest_qua
                       let age_animal = age_animal of pretest_qua
                       let per_acclimatation = per_acclimatation of pretest_qua
                       let date_entree = date_entree of pretest_qua
                       let bool_etude = "V"
                       let bool_quar = "V"
                     end
                   else
                     begin
                       get pretest_cin via num_animal using cle_animal_quar &
                            optional
                       if accessok
                         then
                           begin
                             let age_animal = age_animal of pretest_cin
                             let date_entree = date_entree of pretest_cin
                             let bool_quar = "V"
                           end
                     end

; (MISE_A_JOUR DE TAB_CHANGEMENTS)

                     let code_bidon of tab_changements = code_etude_bidon &
                                                          of elemvariables
                     let code_def of tab_changements = code_etude of elemcommuns
                     let nr_def of tab_changements = nr_animal of elemvariables
                  end
           end
         else
           if etat of elemcommuns = "QUAR"
              then
                begin
                   get cl_code via code_bidon using code_etude of elemcommuns &
```

```
                    optional
                if accessok
                  then
                    begin
                      get pretest_six via num_animal using &
                         (nr_animal of elemvariables + code_def of cl_code)
                      let age_animal = age_animal of pretest_six
                      let date_entree = date_entree of pretest_six
                      let bool_quar = "V"
                    end
                  else
                    begin
                      get pretest_seven via num_animal using &
                         cle_animal optional
                      if accessok
                        then
                          begin
                            let age_animal = age_animal of pretest_seven
                            let date_entree = date_entree of pretest_seven
                            let bool_quar = "V"
                          end
                    end
            end

;( Recherche du fournisseur de l'animal pour l'afficher si on le connait déjà)

   if espece of elemcommuns = "RAT"
      then
        begin
           if etat of elemcommuns = "PRETEST"
             then
               begin
                  get animal via num_animal using cle_animal optional
                  if accessok
                    then
                      begin
                        LET SEXE OF ELEMVARIABLES = SEXE OF ANIMAL
                        get fournisseur via num_fournisseur &
                                        using num_fournisseur of animal
                         let nom_fournisseur of elemvariables = nom_fournisseur &
                                                                of fournisseur
                         let num_fournisseur of elemvariables = num_fournisseur &
                                                                of fournisseur
                         end
               end
             else
                if etat of elemcommuns = "TEST"
                   then
                     begin
                       get animal_bis via num_animal &
                         using cle_nr_lot_code_def optional
                       if accessok
                          then
                            begin
                              LET SEXE OF ELEMVARIABLES = SEXE OF ANIMAL_bis
                              get fournisseur via num_fournisseur &
                                        using num_fournisseur of animal_bis
                              let nom_fournisseur of elemvariables = &
                                        nom_fournisseur of fournisseur
                              let num_fournisseur of elemvariables = &
```

```
                                        num_fournisseur of fournisseur
                    end
                end
  end
else
  if espece of elemcommuns <> "RAT"
     then
       begin
         if etat of elemcommuns = "PRETEST"
           then
             begin
               get animal via num_animal using cle_animal optional
               if accessok
                 then
                   begin
                     LET SEXE OF ELEMVARIABLES = SEXE OF ANIMAL
                     get fournisseur via num_fournisseur &
                                  using num_fournisseur of animal
                     let nom_fournisseur of elemvariables = &
                                  nom_fournisseur  of fournisseur
                     let num_fournisseur of elemvariables = &
                                   num_fournisseur of fournisseur
                   end
                 else
                   begin
                     get animal_ter via num_animal &
                        using cle_nr_def_code_bidon  optional
                     if accessok
                       then
                         begin
                           LET SEXE OF ELEMVARIABLES = SEXE OF ANIMAL_TER
                           get fournisseur via num_fournisseur &
                                    using num_fournisseur of animal_ter
                           let nom_fournisseur of elemvariables = &
                                   nom_fournisseur  of fournisseur
                           let num_fournisseur of elemvariables = &
                                      num_fournisseur of fournisseur
                         end
                   end
             end
               else
                 if etat of elemcommuns = "TEST"
                     then
                        begin
                           get animal_qua via num_animal &
                              using cle_nr_lot_code_def optional
                           if accessok
                             then
                                begin
                                LET SEXE OF ELEMVARIABLES = &
                                                    SEXE OF ANIMAL_qua
                                  get fournisseur via num_fournisseur &
                                      using num_fournisseur of animal_qua
                                  let nom_fournisseur of elemvariables = &
                                       nom_fournisseur of fournisseur
                                  let num_fournisseur of elemvariables = &
                                       num_fournisseur of fournisseur
                                end
                             else
                                begin
```

```
                                get animal_cin via num_animal &
                                    using cle_animal_quar optional
                                if accessok
                                  then
                                    begin
                                      LET SEXE OF ELEMVARIABLES = &
                                          SEXE OF ANIMAL_cin
                                     get fournisseur via num_fournisseur &
                                        using num_fournisseur &
                                            of animal_cin
                                      let nom_fournisseur of elemvariables &
                                        = nom_fournisseur of fournisseur
                                      let num_fournisseur of elemvariables &
                                        = num_fournisseur of fournisseur
                                      end
                              end
                        end
                      else
                        if etat of elemcommuns = "QUAR"
                         then
                          begin
                            get cl_code via code_bidon using &
                               code_etude of elemcommuns optional
                            if accessok
                              then
                               begin
                                 get animal_six via num_animal &
                                 using (nr_animal of elemvariables + &
                                      code_def of cl_code)
                                 LET SEXE OF ELEMVARIABLES = &
                                        SEXE OF ANIMAL_six
                                 get fournisseur via num_fournisseur &
                                    using num_fournisseur of animal_six
                                 let nom_fournisseur of elemvariables = &
                                    nom_fournisseur  of fournisseur
                                 let num_fournisseur of elemvariables = &
                                     num_fournisseur of fournisseur
                               end
                             else
                               begin
                                 get animal_seven via num_animal &
                                    using cle_animal optional
                                 if accessok
                                   then
                                     begin
                                       LET SEXE OF ELEMVARIABLES = &
                                             SEXE OF ANIMAL_seven
                                      get fournisseur via num_fournisseur &
                                             using num_fournisseur &
                                                of animal_seven
                                     let nom_fournisseur of elemvariables = &
                                           nom_fournisseur  of fournisseur
                                     let num_fournisseur of elemvariables = &
                                           num_fournisseur of fournisseur
                                       end
                               end
                          end
                end
end
```

```
procedure PROCESS unite_duree_quarantaine
  begin
    if unite_duree_quarantaine <> "j"
       then
         begin
           get table_mesure via entree using unite_duree_quarantaine
           while retrieving table_mesure
             begin
               if sortie = "j"
                 then
                   begin
                     let unite_duree_quarantaine = "j"
                     let duree = (duree * facteur of table_mesure)
                     display duree
                     break
                   end
             end
        end
  end


Procedure INPUT age_animal
begin
   if matchpattern(fieldtext,"!0")
     then
       if age_animal LE 0
         then
            error "You MUST enter a value"
end

procedure PROCESS unite_age_animal
  begin
    if unite_age_animal <> "j"
       then
         begin
           get table_mesure via entree using unite_age_animal
           while retrieving table_mesure
             begin
               if sortie = "j"
                 then
                   begin
                     let unite_age_animal = "j"
                     let age_animal = (age_animal * facteur of table_mesure)
                     display age_animal
                     break
                   end
             end
        end
  end

Procedure INPUT per_acclimatation
begin
   if matchpattern(fieldtext,"!0")
     then
       if per_acclimatation LE 0
         then
            error "You MUST enter a value"
end
```

```
procedure PROCESS unite_per_acclimatation
  begin
    if unite_per_acclimatation <> "j"
       then
         begin
           get table_mesure via entree using unite_per_acclimatation
           while retrieving table_mesure
             begin
               if sortie = "j"
                 then
                   begin
                     let unite_per_acclimatation = "j"
                     let per_acclimatation = (per_acclimatation * facteur &
                                        of table_mesure)
                     display per_acclimatation
                     break
                   end
             end
         end
  end


Procedure INPUT date_entree
begin
   if matchpattern(fieldtext,"!0")
     then
       if date_entree LE 0
         then
            error "You MUST enter a value"
end


Procedure PROCESS date_entree
begin
  if etat of elemcommuns = "QUAR"
    then
      begin
        if date_debut_etude of elemcommuns <> date_entree of elemvariables
          then error &
"La date d'entrée en quarantaine doit correspondre à la date de début d'étude!"
      end
    else
      begin
        if date_debut_etude of elemcommuns <= date_entree of elemvariables
          then error &
"La date d'entrée en quarantaine doit être inférieure à la date de début étude!"
      end
end


Procedure INPUT nom_fournisseur
begin
  if matchpattern(fieldtext,"!0")
     then
       if nom_fournisseur LE " "
        then
            error "You MUST enter a value"
end
```

```
procedure PROCESS nom_fournisseur
  begin
    get fournisseur via nom_fournisseur using nom_fournisseur of elemvariables &
              optional
    if accessok
       then
         begin
           let num_fournisseur = num_fournisseur of fournisseur
         end
       else
         begin
           get num_max sequential
           let num_fournisseur of num_max = num_fournisseur of num_max + 1
           let num_fournisseur = num_fournisseur of num_max
           let num_fournisseur of fournisseur = num_fournisseur of num_max
           put num_max
           let nom_fournisseur of fournisseur = nom_fournisseur of elemvariables
           put fournisseur
         end
  end


Procedure INPUT vol_administre
begin
  if matchpattern(fieldtext,"!0")
    then
       if vol_administre LE 0
         then
            error "You MUST enter a value"
end

procedure PROCESS unite_vol_administre
  begin
    if unite_vol_administre <> "ml"
       then
         begin
           get table_mesure via entree using unite_vol_administre
           while retrieving table_mesure
             begin
               if sortie = "ml"
                 then
                   begin
                     let unite_vol_administre = "ml"
                     let vol_administre = (vol_administre * facteur &
                                          of table_mesure)
                     display vol_administre
                     break
                   end
             end
        end
  end


BUILD
```

```
screen elements_hypervariables_man &
       receiving elemvariables,d_nr_lot,&
                 d_nr_def , d_code_bidon , &
                 d_code_def , d_date_debut_etude,&
                 tab_changements_put

description of screen &
    "Entrée des éléments variables d'un résultat à l'autre pour un même animal"

file elemvariables master need 1
file elemhypervariables primary
file tab_changements designer

file parambiol reference
file elemcommuns designer
file determinant designer
file elemhypervariables alias elemhypervariables_bis designer

define d_date_debut_etude date
define d_nr_lot char*8
define d_nr_def char*8
define d_code_def char*8
define d_code_bidon char*8

temporary t_resultat date
temporary unite_valeur_res char*8
temporary tab_changements_put char*1
temporary existant char*4
temporary var_num_animal char*16

;*************************************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "E N T R E E  D E S  E L E M E N T S  H Y P E R V A R I A B L E S" &
   centered
skip to 6
align (3,8,38)
;*************************************************************************

FIELD NOM_DET OF ELEMHYPERVARIABLES required &
       lookup on determinant via nom_det using nom_det of elemhypervariables &
       message "Ce determinant n'est pas repertorie !" &
       label "Nom du déterminant"

FIELD NOM_PARAM_BIO OF ELEMHYPERVARIABLES required &
         lookup on parambiol &
         message "Ce paramètre biologique n'est pas repertorie!"

FIELD METHODE_ANALYSE OF ELEMHYPERVARIABLES required
FIELD JOUR_ETUDE OF ELEMHYPERVARIABLES required label "Jour de l'étude"
FIELD DATE_RES OF ELEMHYPERVARIABLES display
FIELD VALEUR_RES OF ELEMHYPERVARIABLES required
align (,,52)
field unite_valeur_res display
align (3,8,20)
skip 1
hilite title underline
title "Commentaire :" at 15 ,3
align (,,1)
```

```
FIELD COMMENTAIRE OF ELEMHYPERVARIABLES noid nolabel

Procedure EDIT jour_etude
  begin
    get elemcommuns sequential
    if etat of elemcommuns = "PRETEST"
     then
       if jour_etude >= 0
        then
          error "La valeur introuduite doit être inférieure à zéro !"
        else
          let t_resultat = date(days(d_date_debut_etude)+jour_etude)
     else
       if (etat of elemcommuns = "TEST") or (etat of elemcommuns = "QUAR")
        then
          if jour_etude <= 0
            then
              error "La valeur introduite doit être supérieure à zéro !"
            else
              let t_resultat = date(days(d_date_debut_etude)+(jour_etude - 1))

;
;nous initialisons ci-dessous le champs num_animal
;
    let num_animal = num_animal of elemvariables
    let date_res = t_resultat
    get elemcommuns sequential
    if etat of elemcommuns = "QUAR"
      then
        begin
          if date_res of elemhypervariables < d_date_debut_etude
           then
            error "La date du résultat est inférieure à la date de début étude!"
        end
      else
         if etat of elemcommuns = "PRETEST"
           then
             begin
             if (date_res of elemhypervariables > date_entree of elemvariables)&
               or (date_res of elemhypervariables > d_date_debut_etude)
              then
                error "La date du résultat est incohérente !"
            end
          else
             if etat of elemcommuns = "TEST"
              then
                begin
                  if (date_res of elemhypervariables &
                       > date_entree of elemvariables)&
                     or (date_res of elemhypervariables < d_date_debut_etude)
              then
                error "La date du résultat est incohérente !"
             end
end

Procedure PROCESS nom_param_bio
begin
  if accessok
   then
     begin
```

```
      get determinant via nom_det &
                      using nom_det of elemhypervariables
      while retrieving determinant
             begin
                if nom_param_bio of elemhypervariables &
                       = nom_param_bio of determinant
                   then
                     begin
                       let unite_valeur_res = unite_mesure of parambiol
                       break
                     end
             end
      if nom_param_bio of elemhypervariables <> nom_param_bio of determinant
        then
          error "Ce paramètre biologique n'est pas rattaché à ce déterminant !"
     end
end


Procedure PREUPDATE
BEGIN
  if not deletedrecord
   then
begin

; TEST POUR VOIR SI CES VALEURS N'ONT PAS ENCORE ETE INTRODUITES

  let var_num_animal = num_animal of elemhypervariables
  get elemhypervariables_bis via num_animal using var_num_animal  &
           optional
  if accessok
   then
     while retrieving elemhypervariables_bis
      begin
        if nom_det of elemhypervariables_bis = nom_det of elemhypervariables
         then
          if nom_param_bio of elemhypervariables_bis = nom_param_bio &
                    of elemhypervariables
            then
             if methode_analyse of elemhypervariables_bis = methode_analyse &
                    of elemhypervariables
               then
                if jour_etude of elemhypervariables_bis = jour_etude &
                    of elemhypervariables
                 then
                  if valeur_res of elemhypervariables_bis = valeur_res &
                   of elemhypervariables
                   then
                   error "Vous avez déjà introduits ces valeurs pour cet animal"
      end

; FIN TEST

  if tab_changements_put = "V"
    then
      begin
        get tab_changements via cle_tab using (d_code_bidon + d_code_def)&
               optional
        if accessok
          then
```

```
          begin
           while retrieving tab_changements
            begin
             if d_nr_lot <> nr_lot of tab_changements
                then
                   let existant = "FAUX"
                else
                  if d_nr_def <> nr_def of tab_changements
                     then
                        let existant = "FAUX"
                     else
                       begin
                         let existant = "VRAI"
                         break
                       end
            end
           if existant = "FAUX"
              then
                begin
                  let nr_lot of tab_changements = d_nr_lot
                  let nr_def of tab_changements = d_nr_def
                  let code_bidon of tab_changements = d_code_bidon
                  let code_def of tab_changements = d_code_def
                  put tab_changements
                end
          end
         else
                begin
                  let nr_lot of tab_changements = d_nr_lot
                  let nr_def of tab_changements = d_nr_def
                  let code_bidon of tab_changements = d_code_bidon
                  let code_def of tab_changements = d_code_def
                  put tab_changements
                end

      end
end

END

BUILD
```

```
screen ecran_poids_menu

FILE POIDS PRIMARY occurs 10
file espece reference
file poids alias poids_bis designer
file poids alias poids_ter designer
file poids alias poids_qua designer
file poidsinter designer
file table_mesure designer
file cl_code designer
file cl_numero designer

temporary etat char*8
temporary espece char*10
temporary code_etude char*8
temporary code_etude_bidon char*8
temporary nr_animal char*8
temporary d_nr_lot char*8
temporary unite_mesure_poids char*8

temporary cle_num_animal_def char*16
temporary poids_trouve char*1
temporary trouve_code char*8 initial " "
temporary code_bidon_trouve char*1

;-----------------------------------------------------------------
draw thick 2,5 to 4,75
skip to 3
hilite title  inverse
title " E N T R E E   D E   D O N N E E S   D E   P O I D S " &
centered
;-----------------------------------------------------------------
skip to 6
align (3,6,30)
hilite id inverse

FIELD ETAT label "Etat [TEST]" required upshift values "TEST","PRETEST", &
                                   "QUAR"
align (,40,60)
FIELD ESPECE lookup on espece via espece using espece &
                message "Cette espece est inconnue !" &
                label "Espece [RAT]" required id same upshift
align (3,6,30)
FIELD CODE_ETUDE_BIDON if espece <> "RAT"  &
                       required  label "Code étude quarantaine" upshift
align (,40,60)
FIELD CODE_ETUDE if etat <> "QUAR" required  label "Code étude" id same &
                                upshift

align (3,6,30)
FIELD D_NR_LOT required label "Numéro de lot" upshift
align (,40,60)
FIELD NR_ANIMAL if etat = "TEST" upshift &
            required label "Numéro d'animal" id same

align(3,6,18)
;FIELD NUM_ANIMAL_TEMP if findmode upshift
hilite title underline
title "      DATE MESURE          VALEUR      UNITE        NUM_ANIMAL      " &
                          at 11,3
```

```
cluster occurs with poids at 12,6
align(6,,9)(,,27)
FIELD DATE_MESURE OF POIDS  nochange required
FIELD VALEUR_POIDS OF POIDS nochange required
align(,,42)
FIELD UNITE_MESURE_POIDS required id same
align(,,53)
FIELD NUM_ANIMAL OF POIDS if findmode and path = 2 nochange id same upshift



Procedure INPUT etat
begin
  if matchpattern(fieldtext,"!0")
    then let fieldtext = "TEST"
end


Procedure INPUT espece
begin
  if matchpattern(fieldtext,"!0")
    then let fieldtext = "RAT"
end


Procedure PROCESS espece
begin
  if etat = "QUAR"
    then
      if espece = "RAT"
        then
          error "On ne peut entrer des données de quarantaine pour un rat !"
end

Procedure PROCESS code_etude
begin
  get cl_code via code_def using code_etude  optional
  if accessok
    then
      begin
        while retrieving cl_code
         begin
          if code_bidon of cl_code <> code_etude_bidon
              then let code_bidon_trouve = "F"
              else
                begin
                  let code_bidon_trouve = "V"
                  break
                end
         end
        if code_bidon_trouve = "F"
           then error "Le code étude bidon ne correspond pas à ce code étude !"
      end
end

Procedure PROCESS d_nr_lot
   begin
     if etat = "QUAR"
       then
```

```
        let cle_num_animal_def = d_nr_lot + code_etude_bidon
      else
        let cle_num_animal_def = d_nr_lot + code_etude

; ( On va rechercher des donnees poids qui pourraient avoir été enregitrées
;   avec un ancien num_animal et si c'est le cas, on les mets à jour )

     if espece <> "RAT"
       then
         if etat <> "QUAR"
           then
             begin
               get cl_code via code_def using code_etude optional
               if accessok
                 then
                   begin
                     let poids_trouve = "F"
                     while retrieving cl_code
                       begin
                         get poids_bis via num_animal using &
                                 (d_nr_lot + code_bidon of cl_code)  optional

                         if accessok
                           then
                             begin
                               let trouve_code = code_bidon of cl_code
                               let poids_trouve = "V"
                             end
                         if poids_trouve = "V"
                           then break
                       end
                     get poids_ter via num_animal using &
                               (d_nr_lot + trouve_code) optional
                     if accessok
                       then
                         begin
                           while retrieving poids_ter
                             begin
                               let num_animal of poids_ter = &
                                       (d_nr_lot + code_etude)
                               put poids_ter
                             end
                         end
                   end
                 else
                   begin
                     get poids_bis via num_animal using &
                              (d_nr_lot + code_etude_bidon) optional
                     if accessok
                       then
                         while retrieving poids_bis
                           begin
                             let num_animal of poids_bis = &
                                     (d_nr_lot + code_etude)
                             put poids_bis
                           end
                   end
             end
           else
             begin
```

```
            get poids_bis via num_animal using (d_nr_lot + "@") &
                                optional
            if accessok
              then
                while retrieving poids_bis
                 if num_animal of poids_bis <> (d_nr_lot + code_etude_bidon)
                  then
                    begin
                      let cle_num_animal_def = num_animal of poids_bis
                      break
                    end
          end
   end


Procedure PROCESS nr_animal
begin
  get cl_numero via nr_lot using d_nr_lot optional
  if accessok
    then
      if nr_def of cl_numero <> nr_animal
        then
          error "Ce numero d'animal ne correspond pas au numéro de lot donné !"

  get cl_numero via nr_def using nr_animal optional
  if accessok
    then
      if nr_lot of cl_numero <> d_nr_lot
        then
         error "Le numero de lot donné ne correspond pas à ce numéro d'animal !"
end

Procedure PROCESS date_mesure
begin
  let num_animal of poids = cle_num_animal_def
  display num_animal of poids
end



Procedure PROCESS unite_mesure_poids
begin
  if unite_mesure_poids <> "gr"
     then
       begin
         get table_mesure via entree using unite_mesure_poids
         while retrieving table_mesure
           if sortie = "gr"
             then
               begin
                 let unite_mesure_poids = "gr"
                 let valeur_poids of poids = &
                              (valeur_poids * facteur of table_mesure)
                 display valeur_poids of poids
                 break
               end
       end

   get poids_qua via num_animal using num_animal of poids optional
   if accessok
```

```
    then
      while retrieving poids_qua
        begin
          if date_mesure of poids_qua = date_mesure of poids &
              and valeur_poids of poids_qua = valeur_poids of poids
            then
              error "Cette valeur a déjà été enregistrée"
        end

  get poidsinter via num_animal using num_animal of poids optional
  if accessok
    then
      while retrieving poidsinter
        begin
          if (valeur_poids of poidsinter = valeur_poids of poids) &
             and (date_mesure of poidsinter = date_mesure of poids)
            then
              error "Cette valeur vient d'être introduite !!"
        end
  let num_animal of poidsinter = num_animal of poids
  let valeur_poids of poidsinter = valeur_poids of poids
  let date_mesure of poidsinter = date_mesure of poids
  put poidsinter
end

Procedure EXIT
begin
  run command &
 "$copy [vmmarc.bd.fichvide]poidsinter.dat [vmmarc.bd]poidsinter.dat"
  run command "$purge poidsinter.dat"
end

BUILD
```

```
screen menu_maj_bd menu

;*****************************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title " M I S E - A - J O U R    D E   L A   B.D." centered
skip to 6
;*****************************************************

align(5,8,30)
hilite id inverse
subscreen animal label "FICHIER ANIMAL" clear all auto
subscreen pretest label "FICHIER PRETEST" clear all auto
subscreen test label "FICHIER TEST" clear all auto
subscreen posttest label "FICHIER POSTTEST" clear all auto
subscreen resultat label "FICHIER RESULTAT" clear all auto
subscreen condexperiment label "FICHIER CONDITIONS EXP." clear all auto
subscreen poids label "FICHIER POIDS" clear all auto
subscreen determinant label "FICHIER DETERMINANT" clear all auto
subscreen animal label "FICHIER ANIMAL" clear all auto
subscreen fournisseur label "FICHIER FOURNISSEUR" clear all auto
subscreen parametre label "FICHIER PARAMETRE BIOL." clear all auto


build
```

```
screen menu_exploitation_bd menu

temporary t_filename char*24
define d_com char*62  = "$crea/fdl=[vmmarc.bd]requete_biol.fdl " + t_filename
define d_delete char*40 = "$delete/nolog "+ t_filename +";*"
define d_assignation char*49 = "$assign/job "+ t_filename + " requete_biol"
define d_deassignation char*27 = "$deassign/job requete_biol "

;*************************************
draw thick 4,5 to 8,75
skip to 6
hilite title blinking inverse
title " E X P L O I T A T I O N      D E      L A   B.D." centered
;*************************************

align (11,14,57)
skip 4
hilite title off
hilite id inverse

hilite title inverse
title "01" at 11,11
hilite title off
title "Statistiques en biologie clinique" at 11,14

skip 2
subscreen menu_stat_poids &
    clear all auto id 2&
    label "Statistiques en toxicologie (Poids)"

Procedure DESIGNER 1
 begin
   let t_filename = "requete_biol" + ascii(systime) +".dat"
   run command d_com
   run command d_assignation
   run screen menu_stat_biol_clin
   run command d_delete
   run command d_deassignation
 end
build
```

```
screen menu_stat_biol_clin

file requete_biol close
file espece reference
file race reference
file fournisseur reference
file table_mesure designer
file parambiol reference
  item operateur_date initial "="
  item operateur_age initial "="

temporary unite_age_animal char*4 initial "SEM"
temporary traitement char initial "Y"
temporary del_fruit_selection char*81

temporary QTP_1 char initial "F"
temporary QTP_2 char initial "F"
temporary QTP_3 char initial "F"
temporary QTP_4 char initial "F"
temporary QTP_5 char initial "F"
temporary QTP_6 char initial "F"
temporary qtp_2_1 char initial "F"
temporary QTP_3_1 char initial "F"
temporary QTP_3_2 char initial "F"
temporary QTP_4_1 char initial "F"
temporary QTP_4_2 char initial "F"
temporary QTP_4_3 char initial "F"
temporary QTP_5_1 char initial "F"
temporary QTP_5_2 char initial "F"
temporary QTP_5_3 char initial "F"
temporary QTP_5_4 char initial "F"
temporary QTP_6_4 char initial "F"
temporary QTP_6_5 char initial "F"
temporary QTP_5_4_FIN char initial "F"
temporary QTP_5_3_FIN char initial "F"
temporary QTP_5_2_FIN char initial "F"
temporary QTP_4_1_FIN char initial "F"
temporary QTP_4_2_FIN char initial "F"
temporary QTP_4_3_FIN char initial "F"
temporary QTP_4_FIN char initial "F"

;*******************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "C R I T E R E S    D E    S E L E C T I O N" centered
skip to 6
align (5,11,45)
;*******************************************

hilite id inverse
hilite display halftone
FIELD ESPECE OF REQUETE_BIOL lookup on espece via espece  &
                    message "Cette espèce n'est pas répertoriée !" &
                    label "Espece"
FIELD RACE OF REQUETE_BIOL lookup on race via race &
                  message "Cette race n'est pas répertoriée !" &
                    label "Race"
FIELD NOM_FOURNISSEUR OF REQUETE_BIOL lookup on fournisseur &
         via nom_fournisseur &
```

```
        message "Ce fournisseur n'est pas répertorié !" &
        label "Nom fournisseur"
FIELD SEXE OF REQUETE_BIOL label "Sexe"
FIELD OPERATEUR_AGE OF REQUETE_BIOL predisplay label "Age animal" &
                          values "<",">","="
align (,,47)
FIELD AGE_ANIMAL OF REQUETE_BIOL nolabel id same
align (,,54)
field unite_age_animal predisplay nolabel id same
align(5,11,45)
FIELD OPERATEUR_DATE OF REQUETE_BIOL predisplay label "Date entrée" &
                          values "<",">","="
align(,,47)
FIELD DATE_ENTREE OF REQUETE_BIOL nolabel id same
align(5,11,45)
FIELD ETAT_ALIMENTAIRE OF REQUETE_BIOL label "Etat alimentaire"
FIELD TYPE_ECHANTILLON OF REQUETE_BIOL label "Type_echantillon"
FIELD NOM_PARAM_BIO OF REQUETE_BIOL lookup on parambiol &
             message "Ce paramètre biologique n'est pas répertorié !" required
FIELD METHODE_ANALYSE OF REQUETE_BIOL label "Méthode analyse"
skip 2

hilite title underline
TITLE "Statistiques:" at 17,5
hilite title off

TITLE "11" at 19,8
TITLE  "Statistiques de biologie clinique" at 19,11

Procedure DESIGNER 11
begin
  if nom_param_bio of requete_biol = ""
  then error "You must first enter values for parameters !"
  else
   begin
     clear screen
     run screen stat_de_bio_clin mode E
     refresh screen
   end
end

Procedure PROCESS unite_age_animal
  begin
    if unite_age_animal <> "j"
       then
         begin
           get table_mesure via entree using unite_age_animal
           while retrieving table_mesure
             begin
               if sortie = "j"
                 then
                   begin
                     let unite_age_animal = "j"
                     let age_animal = (age_animal * facteur of table_mesure)
                     display age_animal
                     break
                   end
             end
       end
  end
```

```
Procedure POSTUPDATE
begin
      do external orga_requete ( espece,race,NOM_fournisseur,sexe,age_animal, &
          date_entree,etat_alimentaire,type_echantillon,nom_param_bio, &
          methode_analyse, &
          QTP_1,QTP_2,QTP_3,QTP_4,QTP_5,QTP_6, &
          qtp_2_1,QTP_3_1,QTP_3_2,QTP_4_1,QTP_4_2,QTP_4_3,QTP_5_1,QTP_5_2, &
          QTP_5_3,QTP_5_4,QTP_6_4,QTP_6_5,QTP_5_4_FIN,QTP_5_3_FIN,QTP_5_2_FIN,&
          QTP_4_1_FIN,QTP_4_2_FIN,QTP_4_3_FIN,QTP_4_FIN)

;************************************************
;mise à blanc du fichier FRUIT_SELECTION_6
;************************************************

run command "$delete fruit_selection_6.dat;*"

let del_fruit_selection = "$copy [vmmarc.bd.fichvide]fruit_selection_6.dat " + &
                          "[vmmarc.bd]fruit_selection_6.dat"
run command del_fruit_selection;

;************************************************
;lancement des QTP true
;************************************************

    if QTP_1 = "T"
      then
       begin
         if operateur_date of requete_biol = "<"
           then
             begin
               clear screen
               run command "QTP auto=qtp_exe:QTP_1_MOINS.qtc" &
               refresh screen
             end
           else
             begin
               if operateur_date of requete_biol = ">"
                 then
                   begin
                     clear screen
                     run command "QTP auto=qtp_exe:QTP_1_PLUS.qtc" &
                     refresh screen
                   end
                else
                  begin
                    clear screen
                    run command "QTP auto=qtp_exe:QTP_1.qtc" &
                    refresh screen
                  end
             end
       end
    if QTP_2 = "T"
      then
       begin
        clear screen
        run command "QTP auto=qtp_exe:QTP_2.qtc" &
        refresh screen
       end
```

```
if QTP_3 = "T"
  then
   begin
    clear screen
    run command "QTP auto=qtp_exe:QTP_3.qtc" &
    refresh screen
   end
if QTP_4 = "T"
  then
   begin
    clear screen
    run command "QTP auto=qtp_exe:QTP_4.qtc" &
    refresh screen
   end
if QTP_2_1 = "T"
  then
   begin
    clear screen
    run command "QTP auto=qtp_exe:QTP_2_1.qtc" &
    refresh screen
   end
if QTP_3_1 = "T"
  then
   begin
    clear screen
    run command "QTP auto=qtp_exe:QTP_3_1.qtc" &
    refresh screen
   end
if QTP_3_2 = "T"
  then
   begin
    clear screen
    run command "QTP auto=qtp_exe:QTP_3_2.qtc" &
    refresh screen
   end
if QTP_4_1 = "T"
  then
   begin
    clear screen
    run command "QTP auto=qtp_exe:QTP_4_1.qtc" &
    refresh screen
   end
if QTP_4_2 = "T"
  then
   begin
    clear screen
    run command "QTP auto=qtp_exe:QTP_4_2.qtc" &
    refresh screen
   end
if QTP_4_3 = "T"
  then
   begin
    clear screen
    run command "QTP auto=qtp_exe:QTP_4_3.qtc" &
    refresh screen
   end
if QTP_5_4 = "T"
  then
   begin
    clear screen
```

```
        run command "QTP auto=qtp_exe:QTP_5_4.qtc" &
        refresh screen
       end
    if QTP_6_4 = "T"
      then
       begin
        clear screen
        run command "QTP auto=qtp_exe:QTP_6_4.qtc" &
        refresh screen
       end
    if QTP_6_5 = "T"
      then
       begin
        clear screen
        run command "QTP auto=qtp_exe:QTP_6_5.qtc" &
        refresh screen
       end
    if QTP_5_4_FIN = "T"
      then
       begin
        clear screen
        run command "QTP auto=qtp_exe:QTP_5_4_FIN.qtc" &
        refresh screen
       end
    if QTP_4_1_FIN = "T"
      then
       begin
        clear screen
        run command "QTP auto=qtp_exe:QTP_4_1_FIN.qtc" &
        refresh screen
       end
    if QTP_4_2_FIN = "T"
      then
       begin
        clear screen
        run command "QTP auto=qtp_exe:QTP_4_2_FIN.qtc" &
        refresh screen
       end
    if QTP_4_3_FIN = "T"
      then
       begin
        clear screen
        run command "QTP auto=qtp_exe:QTP_4_3_FIN.qtc" &
        refresh screen
       end
    if QTP_4_FIN = "T"
      then
       begin
        clear screen
        run command "QTP auto=qtp_exe:QTP_4_FIN.qtc" &
        refresh screen
       end

; (Effacement des fichiers intermédiaires lors de la sélection )

run command "$delete fruit_selection_1.dat;*"
run command "$delete fruit_selection_2.dat;*"
run command "$delete fruit_selection_3.dat;*"
run command "$delete fruit_selection_4.dat;*"
run command "$delete fruit_selection_5.dat;*"
```

```
run command "$delete fruit_pre_selection_5.dat;*"

let del_fruit_selection = "$copy [vmmarc.bd.fichvide]fruit_selection_1.dat " + &
                          "[vmmarc.bd]fruit_selection_1.dat"
run command del_fruit_selection;
let del_fruit_selection = "$copy [vmmarc.bd.fichvide]fruit_selection_2.dat " + &
                          "[vmmarc.bd]fruit_selection_2.dat"
run command del_fruit_selection;
let del_fruit_selection = "$copy [vmmarc.bd.fichvide]fruit_selection_3.dat " + &
                          "[vmmarc.bd]fruit_selection_3.dat"
run command del_fruit_selection;
let del_fruit_selection = "$copy [vmmarc.bd.fichvide]fruit_selection_4.dat " + &
                          "[vmmarc.bd]fruit_selection_4.dat"
run command del_fruit_selection;
let del_fruit_selection = "$copy [vmmarc.bd.fichvide]fruit_selection_5.dat " + &
                          "[vmmarc.bd]fruit_selection_5.dat"
run command del_fruit_selection;

let del_fruit_selection = &
                    "$copy [vmmarc.bd.fichvide]fruit_pre_selection_5.dat " + &
                          "[vmmarc.bd]"
run command del_fruit_selection;


end

BUILD
```

```
screen stat_de_bio_clin menu

file requete_biol designer
file parambiol reference

temporary t_param_determine char*15
temporary ko char*1 initial "F"

;*****************************************
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "S A T I S T I Q U E S   D E   B I O L O G I E   C L I N I Q U E " &
         centered
skip to 8
align (5,11,45)
;*****************************************

field t_param_determine required upshift &
               noid label "Type de statistique :"

hilite title underline
title "A choisir parmi :" at 10,11
hilite title off
title " ESPECE" at 11,30
title " AGE" at 12,30
title " POIDS" at 13,30
title " SEXE" at 14,30
title " TYPE_ALIMENT" at 15,30
title " FOURNISSEUR" at 16,30
title " RACE" at 17,30
title " TYPE INJECTION" at 18,30


Procedure PROCESS t_param_determine
begin
  get requete_biol sequential
  get parambiol via nom_param_bio using nom_param_bio of requete_biol
  let unite_mesure of requete_biol = unite_mesure of parambiol
  if t_param_determine = "AGE"
    then
      begin
        let typestat of requete_biol = "TA"
        clear screen
        run command "quiz auto=quiz_exe:traitement_age.qzc"
        refresh screen
      end
  else
  if t_param_determine = "POIDS"
    then
      begin
        let typestat of requete_biol = "TP"
        clear screen
        run command "quiz auto=quiz_exe:traitement_poids.qzc"
        refresh screen
      end
  else
  if t_param_determine = "SEXE"
    then
      begin
```

```
      let typestat of requete_biol = "TS"
      clear screen
      run command "quiz auto=quiz_exe:traitement_sexe.qzc"
      refresh screen
    end
else
if t_param_determine = "TYPE_ALIMENT"
  then
    begin
      let typestat of requete_biol = "TAL"
      clear screen
      run command "quiz auto=quiz_exe:traitement_type_aliment.qzc"
      refresh screen
    end
else
if t_param_determine = "FOURNISSEUR"
  then
    begin
      let typestat of requete_biol = "TF"
      clear screen
      run command "quiz auto=quiz_exe:traitement_fournisseur.qzc"
      refresh screen
    end
else
if t_param_determine = "RACE"
  then
    begin
      let typestat of requete_biol = "TR"
      clear screen
      run command "quiz auto=quiz_exe:traitement_race.qzc"
      refresh screen
    end
else
if t_param_determine = "ESPECE"
  then
    begin
      let typestat of requete_biol = "ESP"
      clear screen
      run command "quiz auto=quiz_exe:traitement_espece.qzc"
      refresh screen
    end
else
if t_param_determine = "TYPE_INJECTION"
  then
    begin
      let typestat of requete_biol = "TI"
      clear screen
      run command "quiz auto=quiz_exe:traitement_type_injection.qzc"
      refresh screen
    end
  else
   begin
     error "Ce facteur est inconnu !"
     let ko = "V"
   end

; ( Lancement du rapport qui construit le fichier "inforecherche" nécessaire
;   pour les statistiques de biologie clinique avec RS1 )
;
  If ko = "F"
```

```
    then
      begin
        put requete_biol
        clear screen
        run command "quiz auto=quiz_exe:prepa_inforecherche.qzc"
        refresh screen
      end
end


build
```

```
screen menu_stat_poids menu

temporary temoin_choix integer

;-----------------------------------------------------------------
draw thick 4,5 to 6,75
skip to 5
hilite title  inverse
title "S T A T I S T I Q U E S   S U R   L E S   P O I D S " &
centered
;-----------------------------------------------------------------

align (11,14,60)
skip 4

hilite title inverse
title "1"
hilite title off
Title "Evolution du POIDS en cours d'étude"  at 10,14
skip 1

hilite title inverse
title "2"
hilite title off
Title "Evolution du POIDS en cours d'étude suivant le sexe " at 12,14
skip 1

hilite title inverse
title "3"
hilite title off
Title "Evolution du POIDS en fonction de l'AGE" at 14,14
skip 1

hilite title inverse
title "4"
hilite title off
Title "Evolution du POIDS en fonction de l'AGE suivant la RACE" at 16,14
skip 1

hilite title inverse
title "5"
hilite title off
Title "Evolution du POIDS en fonction de l'AGE suivant le SEXE" at 18,14
skip 1


Procedure DESIGNER 1
  begin
    clear screen
    run command "QUIZ auto=quiz_exe: ? "
    refresh screen
  end
Procedure DESIGNER 2
  begin
    clear screen
    run command "QUIZ auto=quiz_exe: ? "
    refresh screen
  end
Procedure DESIGNER 3
  begin
```

```
    clear screen
    let temoin_choix = 3
    run screen Evolution_poids_age_menu passing temoin_choix
    refresh screen
  end
Procedure DESIGNER 4
  begin
    clear screen
    let temoin_choix = 4
    run screen Evolution_poids_age_race_menu passing temoin_choix
    refresh screen
  end
Procedure DESIGNER 5
  begin
    clear screen
    let temoin_choix = 5
    run screen Evolution_poids_age_sexe_menu passing temoin_choix
    refresh screen
  end

build
```

```
screen menu_ajout_general menu

temporary t_choix_utilisateur char*1 initial "d"

;-----------------------------------------------------------------
hilite title inverse
draw thick 4,5 to 6,75
skip to 5
title "MENU GENERAL POUR L'AJOUT D'UN NOUVEAU DETERMINANT OU PARAMETRE" &
centered
;-----------------------------------------------------------------

skip 4
hilite title off
title "Pour ajouter un nouveau déterminant (D)" at 11,11
skip 3
title "Pour ajouter un nouveau paramètre (P)" at 14,11
skip 5
align (,11,25)
field t_choix_utilisateur noid label "Choix (D/P):" values "D", "P",&
                                                          "p", "d"
subscreen ajout_determinant   clear all  nolabel noid &
                                   if t_choix_utilisateur = "d" or &
                                      t_choix_utilisateur = "D"
subscreen  ajout_parametre     clear all  nolabel noid &
                                   if t_choix_utilisateur = "p" &
                                      or t_choix_utilisateur = "P"
build
```

```
screen ajout_determinant

file determinant primary
file determinant reference alias determinant1
    select if nom_det = nom_det of determinant
file parambiol secondary

temporary t_nom_det char*25
temporary creation_parametre char*3 initial "OUI"
temporary t_nom_param_bio char*25

;-------------------------------------------------------------------
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "A J O U T   D ' U N   N O U V E A U   D E T E R M I N A N T" &
      centered
skip to 6
align (3,8,35)
;-------------------------------------------------------------------


field nom_det of determinant required    label "Nom du déterminant"
skip 3

field nom_param_bio of determinant required    label "Nom du paramètre"
skip 3

field unite_mesure of parambiol required &
       if creation_parametre = "OUI" &
       label "Unité de mesure"


procedure EDIT nom_param_bio
   begin
     let t_nom_param_bio = fieldtext
     get determinant1 via nom_param_bio using t_nom_param_bio optional
     if accessok
        then
           error "Erreur - ce paramètre est déjà associé à ce déterminant !"
   end

Procedure PROCESS nom_param_bio
  begin
    let nom_param_bio of parambiol = t_nom_param_bio
    get parambiol via nom_param_bio optional
        if accessok
           then
             let creation_parametre = "NON"
           else
             begin
               let nom_param_bio of parambiol = t_nom_param_bio
               let creation_parametre = "OUI"
             end
  end


Procedure update
   begin
      if creation_parametre = "OUI"
```

```
          then
            begin
              put parambiol
            end
        put determinant
    end

build
```

```
screen ajout_parametre

file parambiol primary
file determinant secondary

;-------------------------------------------------------------------------
hilite title inverse
draw thick 2,5 to 4,75
skip to 3
title "A J O U T   D ' U N   N O U V E A U   P A R A M E T R E " &
      centered
skip to 6
align (3,8,35)
;-------------------------------------------------------------------------

field nom_det of determinant lookup on determinant &
                             message "Erreur - Ce déterminant n'existe pas !" &
                             required &
                             label "Nom du déterminant"
skip 3

field nom_param_bio of parambiol lookup noton parambiol &
                                 message "Erreur - Ce paramètre existe déjà !" &
                                 required &
                                 label "Nom du paramètre"
skip 3

field unite_mesure of parambiol required   label "Unité de mesure"

procedure PROCESS unite_mesure
   begin
     let nom_param_bio of determinant = nom_param_bio of parambiol
   end

build
```

```
screen menu_modif_tables menu

;-----------------------------------------------------------------
hilite title inverse
draw thick 4,5 to 6,75
skip to 5
title "MENU GENERAL POUR LA MISE-A-JOUR DES TABLES DES VALEURS PERMISES" &
centered
;-----------------------------------------------------------------

align(11,14,57)
skip 2
hilite id inverse
subscreen vehicule_utilise &
         clear all auto &
         label "MAJ de la table VEHICULE UTILISE"
skip 2
subscreen type_echantillon &
         clear all auto &
         label "MAJ de la table TYPE ECHANTILLON"
skip 2
subscreen voie_injection &
         clear all auto &
         label "MAJ de la table VOIE D'INJECTION"
skip 2
subscreen type_aliment &
          clear all auto &
          label "MAJ de la table TYPE ALIMENT"

skip 2
subscreen espece &
         clear all auto &
         label "MAJ de la table ESPECE"

skip 2
subscreen race &
         clear all auto &
         label "MAJ de la table RACE"

build
```

TRAITEMENTS BATCH (QTP)

```
run garnissage_quar
global temporary gt_code_etude char*8
global temporary gt_date_debut_etude date
global temporary gt_type_aliment char*20
global temporary gt_race char*20
global temporary gt_espece char*10
global temporary gt_etat char*8
global temporary gt_voie_injection char*20
global temporary gt_type_echantillon char*20
global temporary gt_etat_alimentaire char*6
global temporary gt_vehicule_utilise char*25
global temporary gt_num_cond_res integer*10
global temporary gt_num_fournisseur integer*6


;*****************************
request elemcommuns
;*****************************

  access elemcommuns link code_etude of elemcommuns &
                     to code_etude of etudes optional
     item gt_code_etude = code_etude of elemcommuns
     item gt_date_debut_etude = date_debut_etude of elemcommuns
     item gt_type_aliment = type_aliment of elemcommuns
     item gt_race = race of elemcommuns
     item gt_espece = espece of elemcommuns
     item gt_etat = etat of elemcommuns
     item gt_voie_injection = voie_injection of elemcommuns
     item gt_type_echantillon = type_echantillon of elemcommuns
     item gt_etat_alimentaire = etat_alimentaire of elemcommuns
     item gt_vehicule_utilise = vehicule_utilise of elemcommuns

  output etudes alias etudesbis add if not record etudes exists
     item code_etude = code_etude of elemcommuns
     item date_debut_etude = date_debut_etude of elemcommuns

;**********************
request pre_int_cond_exp
;**********************

  access elemcommuns link type_echantillon of elemcommuns to type_echantillon &
   of condexperiment
   select condexperiment if vehicule_utilise = gt_vehicule_utilise &
             and etat_alimentaire = gt_etat_alimentaire &
             and voie_injection = gt_voie_injection

  output pre_int_cond_exp add
    item num_cond_res of pre_int_cond_exp final num_cond_res of condexperiment
    item methode_analyse of pre_int_cond_exp final methode_analyse &
                                                   of condexperiment
    item vol_administre of pre_int_cond_exp final vol_administre &
                                                  of condexperiment

;**************************
request acces_elemvariables
;**************************

  access elemvariables &
         link num_animal of elemvariables to num_animal of animal optional &
        and  num_animal of elemvariables to num_animal of pretest optional &
```

```
      and  num_animal of elemvariables to num_animal of test optional &
      and vol_administre_char of elemvariables to vol_administre_char &
                                          of pre_int_cond_exp optional

  output animal alias animalbis  add  if not record animal exists
     item num_animal final num_animal of elemvariables
     item num_fournisseur  final num_fournisseur of elemvariables
     item sexe  final sexe of elemvariables
     item espece  final gt_espece
     item race  final gt_race


  output pretest add update if not record pretest exists &
                         and (gt_etat = "PRETEST" or gt_etat = "TEST")
     item num_animal of pretest final num_animal of elemvariables
     item date_entree of pretest final date_entree of elemvariables
     item duree of pretest final duree of elemvariables
     item age_animal of pretest final age_animal of elemvariables
     item per_acclimatation of pretest final per_acclimatation of elemvariables

  output test add update if not record test exists &
                       and gt_etat = "TEST"
     item num_animal of test final num_animal of elemvariables
     item type_aliment of test final gt_type_aliment


  output INT_COND_EXP add if record pre_int_cond_exp exists
     item num_cond_res of INT_COND_EXP final num_cond_res &
                                 of pre_int_cond_exp
     item methode_analyse of INT_COND_EXP final methode_analyse &
                                          of pre_int_cond_exp
     item num_animal of INT_COND_EXP final num_animal of elemvariables



;***************
request num_max
;***************

   access num_max
      item gt_num_cond_res initial num_cond_res of num_max
      item gt_num_fournisseur final num_fournisseur of num_max
   subfile ancien_num_max keep include num_cond_res of num_max


;****************************
request elemhypervariables
;****************************

   access elemvariables LINK num_animal of elemvariables &
                        TO num_animal of elemhypervariables &
      link( methode_analyse of elemhypervariables + num_animal &
            of elemhypervariables) to methode_animal &
            of int_cond_exp optional &
      link  num_animal of elemhypervariables + nom_param_bio &
            of elemhypervariables + num_cond_res_char of int_cond_exp &
               + date_res_char of elemhypervariables &
                 to cle_tgv of resultat optional

   sort on methode_analyse of elemhypervariables &
```

```
      on vol_administre of elemvariables &
      on num_animal

   define nouveau char*25 = methode_analyse of elemhypervariables
   define vol_administre_nouveau char*8 = vol_administre_char of elemvariables

   temporary ancien char*25
   temporary vol_administre_ancien char*8

   temporary num_cond_res_precedent integer*10
       item  num_cond_res_precedent = gt_num_cond_res
   item gt_num_cond_res = (gt_num_cond_res + 1) &
            if not record int_cond_exp exists and (ancien <> nouveau &
                        or vol_administre_ancien <> vol_administre_nouveau)

   item ancien = nouveau
   item vol_administre_ancien = vol_administre_nouveau

   output condexperiment add  if not record int_cond_exp exists &
           and gt_num_cond_res <> num_cond_res_precedent
      item num_cond_res final gt_num_cond_res
      item voie_injection final gt_voie_injection
      item vol_administre final vol_administre of elemvariables
      item type_echantillon final gt_type_echantillon
      item methode_analyse final methode_analyse of elemhypervariables
      item vehicule_utilise final gt_vehicule_utilise
      item etat_alimentaire final gt_etat_alimentaire

   output resultat alias resultatter add if not record int_cond_exp exists
      item date_res final date_res of elemhypervariables
      item num_cond_res final gt_num_cond_res
      item valeur_res final valeur_res of elemhypervariables
      item etat final gt_etat
      item commentaire final commentaire of elemhypervariables
      item nom_param_bio final nom_param_bio of elemhypervariables
      item num_animal final num_animal of elemhypervariables


   output resultat alias resultatbis add  &
          if record int_cond_exp exists &
          and not record resultat exists &
          and (valeur_res of resultat <> valeur_res of elemhypervariables)

      item date_res final date_res of elemhypervariables
      item num_cond_res final num_cond_res of int_cond_exp
      item valeur_res final valeur_res of elemhypervariables
      item etat final gt_etat
      item commentaire final commentaire of elemhypervariables
      item nom_param_bio final nom_param_bio of elemhypervariables
      item num_animal final num_animal of elemhypervariables



;*****************************
request MISE_A_JOUR
;*****************************

access *ANCIEN_NUM_MAX link num_cond_res of ancien_num_max to &
                       num_cond_res of num_max
```

```
output num_max update if num_cond_res of num_max <> gt_num_cond_res
     item num_cond_res of num_max final gt_num_cond_res
     item num_fournisseur of num_max final gt_num_fournisseur


;*****************************
request EFFACAGE
;*****************************

access int_cond_exp
output int_cond_exp delete

;*****************************
request EFFACAGE_BIS
;*****************************

access pre_int_cond_exp
output pre_int_cond_exp delete

build
```

```
run garnissage_nr_pre_quar

global temporary gt_code_etude char*8
global temporary gt_date_debut_etude date
global temporary gt_type_aliment char*20
global temporary gt_race char*20
global temporary gt_espece char*10
global temporary gt_etat char*8
global temporary gt_voie_injection char*20
global temporary gt_type_echantillon char*20
global temporary gt_etat_alimentaire char*6
global temporary gt_vehicule_utilise char*25
global temporary gt_num_cond_res integer*10
global temporary gt_num_fournisseur integer*6


;*****************************
request elemcommuns
;*****************************

  access elemcommuns link code_etude of elemcommuns &
                     to code_etude of etudes optional
     item gt_code_etude = code_etude of elemcommuns
     item gt_date_debut_etude = date_debut_etude of elemcommuns
     item gt_type_aliment = type_aliment of elemcommuns
     item gt_race = race of elemcommuns
     item gt_espece = espece of elemcommuns
     item gt_etat = etat of elemcommuns
     item gt_voie_injection = voie_injection of elemcommuns
     item gt_type_echantillon = type_echantillon of elemcommuns
     item gt_etat_alimentaire = etat_alimentaire of elemcommuns
     item gt_vehicule_utilise = vehicule_utilise of elemcommuns

  output etudes alias etudesbis add if not record etudes exists
     item code_etude = code_etude of elemcommuns
     item date_debut_etude = date_debut_etude of elemcommuns

;**********************
request pre_int_cond_exp
;**********************

  access elemcommuns link type_echantillon of elemcommuns to type_echantillon &
   of condexperiment
   select condexperiment if vehicule_utilise = gt_vehicule_utilise &
             and etat_alimentaire = gt_etat_alimentaire &
             and voie_injection = gt_voie_injection

  output pre_int_cond_exp add
    item num_cond_res of pre_int_cond_exp final num_cond_res of condexperiment
    item methode_analyse of pre_int_cond_exp final methode_analyse &
                                                   of condexperiment
    item vol_administre of pre_int_cond_exp final vol_administre &
                                                  of condexperiment

;**************************
request acces_elemvariables
;**************************

  access elemvariables &
         link (nr_animal of elemvariables + code_etude_bidon of elmvariables ) &
```

```
            to num_animal of animal optional &
      and (nr_animal of elemvariables + code_etude_bidon of elemvariables ) &
           to num_animal of pretest optional &

;       and  num_animal of elemvariables to num_animal of test optional &

      and vol_administre_char of elemvariables to vol_administre_char &
                                              of pre_int_cond_exp optional &
      and num_animal of elemvariables to num_animal of animal &
         alias animal_2 optional &
      and num_animal of elemvariables to num_animal of pretest &
                alias pretest_2 optional


  output animal alias animalbis  add  if not record animal exists and &
                                       not record animal_2 exists
     item num_animal final num_animal of elemvariables
     item num_fournisseur  final num_fournisseur of elemvariables
     item sexe  final sexe of elemvariables
     item espece  final gt_espece
     item race  final gt_race

  output animal  update if record animal exists

     item num_animal final num_animal of elemvariables
     item num_fournisseur  final num_fournisseur of elemvariables
     item sexe  final sexe of elemvariables
     item espece  final gt_espece
     item race  final gt_race


  output pretest alias pretest_bis add update if not record pretest exists &
                         and (gt_etat = "PRETEST" or gt_etat = "TEST") &
                         and not record pretest_2 exists
     item num_animal final num_animal of elemvariables
     item date_entree final date_entree of elemvariables
     item duree final duree of elemvariables
     item age_animal final age_animal of elemvariables
     item per_acclimatation final per_acclimatation of elemvariables

  output pretest update if record pretest exists &
                         and (gt_etat = "PRETEST" or gt_etat = "TEST")
     item num_animal of pretest final num_animal of elemvariables
     item date_entree of pretest final date_entree of elemvariables
     item duree of pretest final duree of elemvariables
     item age_animal of pretest final age_animal of elemvariables
     item per_acclimatation of pretest final per_acclimatation of elemvariables

;  output test add update if not record test exists &
;                        and gt_etat = "TEST"
;      item num_animal of test final num_animal of elemvariables
;      item type_aliment of test final gt_type_aliment
;
;  output test alias test_bis  update if record test exists &
;                        and gt_etat = "TEST"
;      item num_animal of test final num_animal of elemvariables
;      item type_aliment of test final gt_type_aliment


  output INT_COND_EXP add if record pre_int_cond_exp exists
```

```
    item num_cond_res of INT_COND_EXP final num_cond_res &
                                of pre_int_cond_exp
    item methode_analyse of INT_COND_EXP final methode_analyse &
                                                of pre_int_cond_exp
    item num_animal of INT_COND_EXP final num_animal of elemvariables


;***************
request num_max
;***************

   access num_max
      item gt_num_cond_res initial num_cond_res of num_max
      item gt_num_fournisseur final num_fournisseur of num_max
   subfile ancien_num_max keep include num_cond_res of num_max


;****************************
request elemhypervariables
;****************************

   access elemvariables LINK num_animal of elemvariables &
                        TO num_animal of elemhypervariables &
      link( methode_analyse of elemhypervariables + num_animal &
            of elemhypervariables) to methode_animal &
            of int_cond_exp optional &
      link  num_animal of elemhypervariables + nom_param_bio &
            of elemhypervariables + num_cond_res_char of int_cond_exp &
               + date_res_char of elemhypervariables &
                 to cle_tgv of resultat optional

   sort on methode_analyse of elemhypervariables &
        on vol_administre of elemvariables &
        on num_animal

   define nouveau char*25 = methode_analyse of elemhypervariables
   define vol_administre_nouveau char*8 = vol_administre_char of elemvariables

   temporary ancien char*25
   temporary vol_administre_ancien char*8

   temporary num_cond_res_precedent integer*10
       item  num_cond_res_precedent = gt_num_cond_res
   item gt_num_cond_res = (gt_num_cond_res + 1) &
             if not record int_cond_exp exists and (ancien <> nouveau &
                         or vol_administre_ancien <> vol_administre_nouveau)

   item ancien = nouveau
   item vol_administre_ancien = vol_administre_nouveau

   output condexperiment add  if not record int_cond_exp exists &
           and gt_num_cond_res <> num_cond_res_precedent
      item num_cond_res final gt_num_cond_res
      item voie_injection final gt_voie_injection
      item vol_administre final vol_administre of elemvariables
      item type_echantillon final gt_type_echantillon
      item methode_analyse final methode_analyse of elemhypervariables
      item vehicule_utilise final gt_vehicule_utilise
      item etat_alimentaire final gt_etat_alimentaire
```

```
   output resultat alias resultatter add if not record int_cond_exp exists
      item date_res final date_res of elemhypervariables
      item num_cond_res final gt_num_cond_res
      item valeur_res final valeur_res of elemhypervariables
      item etat final gt_etat
      item commentaire final commentaire of elemhypervariables
      item nom_param_bio final nom_param_bio of elemhypervariables
      item num_animal final num_animal of elemhypervariables


   output resultat alias resultatbis add  &
          if record int_cond_exp exists &
          and not record resultat exists &
          and (valeur_res of resultat <> valeur_res of elemhypervariables)

      item date_res final date_res of elemhypervariables
      item num_cond_res final num_cond_res of int_cond_exp
      item valeur_res final valeur_res of elemhypervariables
      item etat final gt_etat
      item commentaire final commentaire of elemhypervariables
      item nom_param_bio final nom_param_bio of elemhypervariables
      item num_animal final num_animal of elemhypervariables


;*****************************
request MAJ_ANCIEN_RESULTATS
;*****************************

access elemvariables link (nr_animal of elemvariables + &
                           code_etude_bidon of elemvariables) &
                       to num_animal of resultat

output resultat alias resultat_ancien update if record resultat exists

   item num_animal final num_animal of elemvariables


;*****************************
request MISE_A_JOUR
;*****************************

access *ANCIEN_NUM_MAX link num_cond_res of ancien_num_max to &
                       num_cond_res of num_max

output num_max update if num_cond_res of num_max <> gt_num_cond_res
     item num_cond_res of num_max final gt_num_cond_res
     item num_fournisseur of num_max final gt_num_fournisseur


;*****************************
request EFFACAGE
;*****************************

access int_cond_exp
output int_cond_exp delete

;*****************************
request EFFACAGE_BIS
;*****************************
```

```
access pre_int_cond_exp
output pre_int_cond_exp delete

;*****************************
request maj_cl_code
;*****************************

access tab_changements LINK code_def of tab_changements &
                          TO code_def of cl_code optional

sorted on cle_tab

output cl_code alias cl_code_bis add at start of cle_tab &
                     if (not record cl_code exists) &
      or (record cl_code exists and code_bidon <> code_bidon of tab_changements)

   item code_def final code_def of tab_changements
   item code_bidon final code_bidon of tab_changements

build
```

```
run garnissage_nr_test_quar

global temporary gt_code_etude char*8
global temporary gt_date_debut_etude date
global temporary gt_type_aliment char*20
global temporary gt_race char*20
global temporary gt_espece char*10
global temporary gt_etat char*8
global temporary gt_voie_injection char*20
global temporary gt_type_echantillon char*20
global temporary gt_etat_alimentaire char*6
global temporary gt_vehicule_utilise char*25
global temporary gt_num_cond_res integer*10
global temporary gt_num_fournisseur integer*6


;*****************************
request elemcommuns
;*****************************

  access elemcommuns link code_etude of elemcommuns &
                     to code_etude of etudes optional
     item gt_code_etude = code_etude of elemcommuns
     item gt_date_debut_etude = date_debut_etude of elemcommuns
     item gt_type_aliment = type_aliment of elemcommuns
     item gt_race = race of elemcommuns
     item gt_espece = espece of elemcommuns
     item gt_etat = etat of elemcommuns
     item gt_voie_injection = voie_injection of elemcommuns
     item gt_type_echantillon = type_echantillon of elemcommuns
     item gt_etat_alimentaire = etat_alimentaire of elemcommuns
     item gt_vehicule_utilise = vehicule_utilise of elemcommuns

  output etudes alias etudesbis add if not record etudes exists
     item code_etude = code_etude of elemcommuns
     item date_debut_etude = date_debut_etude of elemcommuns

;**********************
request pre_int_cond_exp
;**********************

  access elemcommuns link type_echantillon of elemcommuns to type_echantillon &
   of condexperiment
   select condexperiment if vehicule_utilise = gt_vehicule_utilise &
             and etat_alimentaire = gt_etat_alimentaire &
             and voie_injection = gt_voie_injection

  output pre_int_cond_exp add
    item num_cond_res of pre_int_cond_exp final num_cond_res of condexperiment
    item methode_analyse of pre_int_cond_exp final methode_analyse &
                                                   of condexperiment
    item vol_administre of pre_int_cond_exp final vol_administre &
                                                  of condexperiment

;***************************
request acces_elemvariables
;***************************

  access elemvariables &
        link (nr_lot of elemvariables + code_etude_bidon of elemvariables ) &
```

```
              to num_animal of animal optional &
      and (nr_lot of elemvariables + code_etude_bidon of elemvariables ) &
              to num_animal of pretest optional &
      and  num_animal of elemvariables to num_animal of test optional &
      and vol_administre_char of elemvariables to vol_administre_char &
                                          of pre_int_cond_exp optional &
      and num_animal of elemvariables to num_animal of animal &
           alias animal_2 optional &
      and num_animal of elemvariables to num_animal of pretest &
           alias pretest_2 optional

  output animal alias animalbis  add  if not record animal exists &
          and not record animal_2 exists
     item num_animal of animalbis final num_animal of elemvariables
     item num_fournisseur of animalbis final num_fournisseur of elemvariables
     item sexe of animalbis final sexe of elemvariables
     item espece of animalbis final gt_espece
     item race of animalbis final gt_race

  output animal update if record animal exists
     item num_animal of animal final num_animal of elemvariables
     item num_fournisseur of animal final num_fournisseur of elemvariables
     item sexe of animal final sexe of elemvariables
     item espece of animal final gt_espece
     item race of animal final gt_race


  output pretest alias pretest_bis add update if not record pretest exists &
                        and (gt_etat = "PRETEST" or gt_etat = "TEST") &
                        and not record pretest_2 exists
     item num_animal of pretest_bis final num_animal of elemvariables
     item date_entree of pretest_bis final date_entree of elemvariables
     item duree of pretest_bis final duree of elemvariables
     item age_animal of pretest_bis final age_animal of elemvariables
     item per_acclimatation of pretest_bis final per_acclimatation of &
                                        elemvariables

  output pretest update if  record pretest exists &
                        and (gt_etat = "PRETEST" or gt_etat = "TEST")
     item num_animal of pretest final num_animal of elemvariables
     item date_entree of pretest final date_entree of elemvariables
     item duree of pretest final duree of elemvariables
     item age_animal of pretest final age_animal of elemvariables
     item per_acclimatation of pretest final per_acclimatation of elemvariables

  output test alias test_bis add update if not record test exists &
                       and gt_etat = "TEST"
     item num_animal final num_animal of elemvariables
     item type_aliment final gt_type_aliment

;  output test update if record test exists &
;                        and gt_etat = "TEST"
;      item num_animal of test final num_animal of elemvariables
;      item type_aliment of test final gt_type_aliment


  output INT_COND_EXP add if record pre_int_cond_exp exists
     item num_cond_res of INT_COND_EXP final num_cond_res &
                                  of pre_int_cond_exp
     item methode_analyse of INT_COND_EXP final methode_analyse &
```

```
                                                    of pre_int_cond_exp
    item num_animal of INT_COND_EXP final num_animal of elemvariables


;***************
request num_max
;***************

   access num_max
      item gt_num_cond_res initial num_cond_res of num_max
      item gt_num_fournisseur final num_fournisseur of num_max
   subfile ancien_num_max keep include num_cond_res of num_max


;***************************
request elemhypervariables
;***************************

   access elemvariables LINK num_animal of elemvariables &
                        TO num_animal of elemhypervariables &
      link( methode_analyse of elemhypervariables + num_animal &
            of elemhypervariables) to methode_animal &
            of int_cond_exp optional &
      link  num_animal of elemhypervariables + nom_param_bio &
            of elemhypervariables + num_cond_res_char of int_cond_exp &
               + date_res_char of elemhypervariables &
                 to cle_tgv of resultat optional

   sort on methode_analyse of elemhypervariables &
        on vol_administre of elemvariables &
        on num_animal

   define nouveau char*25 = methode_analyse of elemhypervariables
   define vol_administre_nouveau char*8 = vol_administre_char of elemvariables

   temporary ancien char*25
   temporary vol_administre_ancien char*8

   temporary num_cond_res_precedent integer*10
       item  num_cond_res_precedent = gt_num_cond_res
   item gt_num_cond_res = (gt_num_cond_res + 1) &
             if not record int_cond_exp exists and (ancien <> nouveau &
                          or vol_administre_ancien <> vol_administre_nouveau)

   item ancien = nouveau
   item vol_administre_ancien = vol_administre_nouveau

   output condexperiment add  if not record int_cond_exp exists &
           and gt_num_cond_res <> num_cond_res_precedent
      item num_cond_res final gt_num_cond_res
      item voie_injection final gt_voie_injection
      item vol_administre final vol_administre of elemvariables
      item type_echantillon final gt_type_echantillon
      item methode_analyse final methode_analyse of elemhypervariables
      item vehicule_utilise final gt_vehicule_utilise
      item etat_alimentaire final gt_etat_alimentaire

   output resultat alias resultatter add if not record int_cond_exp exists
      item date_res final date_res of elemhypervariables
```

```
      item num_cond_res final gt_num_cond_res
      item valeur_res final valeur_res of elemhypervariables
      item etat final gt_etat
      item commentaire final commentaire of elemhypervariables
      item nom_param_bio final nom_param_bio of elemhypervariables
      item num_animal final num_animal of elemhypervariables


   output resultat alias resultatbis add  &
          if record int_cond_exp exists &
          and not record resultat exists &
          and (valeur_res of resultat <> valeur_res of elemhypervariables)

      item date_res final date_res of elemhypervariables
      item num_cond_res final num_cond_res of int_cond_exp
      item valeur_res final valeur_res of elemhypervariables
      item etat final gt_etat
      item commentaire final commentaire of elemhypervariables
      item nom_param_bio final nom_param_bio of elemhypervariables
      item num_animal final num_animal of elemhypervariables


;*****************************
request MAJ_ANCIEN_RESULTATS
;*****************************

access elemvariables link (nr_lot of elemvariables + &
                           code_etude_bidon of elemvariables) &
                       to num_animal of resultat

output resultat alias resultat_ancien update if record resultat exists

   item num_animal final num_animal of elemvariables


;*****************************
request MISE_A_JOUR
;*****************************

access *ANCIEN_NUM_MAX link num_cond_res of ancien_num_max to &
                       num_cond_res of num_max

output num_max update if num_cond_res of num_max <> gt_num_cond_res
     item num_cond_res of num_max final gt_num_cond_res
     item num_fournisseur of num_max final gt_num_fournisseur


;*****************************
request EFFACAGE
;*****************************

access int_cond_exp
output int_cond_exp delete

;*****************************
request EFFACAGE_BIS
;*****************************

access pre_int_cond_exp
output pre_int_cond_exp delete
```

```
;*****************************
request maj_cl_code
;*****************************

access tab_changements LINK code_def of tab_changements &
                          TO code_def of cl_code optional

sorted on cle_tab

output cl_code alias cl_code_bis add at start of cle_tab &
                     if (not record cl_code exists) &
      or (record cl_code exists and code_bidon <> code_bidon of tab_changements)

   item code_def final code_def of tab_changements
   item code_bidon final code_bidon of tab_changements

;*****************************
request maj_cl_numero
;*****************************

access tab_changements LINK nr_lot of tab_changements &
                          TO nr_lot of cl_numero optional

output cl_numero alias cl_numero_bis add if not record cl_numero exists

   item nr_def final nr_def of tab_changements
   item nr_lot final nr_lot of tab_changements

build
```

```
run garnissage_rat_test

global temporary gt_code_etude char*8
global temporary gt_date_debut_etude date
global temporary gt_type_aliment char*20
global temporary gt_race char*20
global temporary gt_espece char*10
global temporary gt_etat char*8
global temporary gt_voie_injection char*20
global temporary gt_type_echantillon char*20
global temporary gt_etat_alimentaire char*6
global temporary gt_vehicule_utilise char*25
global temporary gt_num_cond_res integer*10
global temporary gt_num_fournisseur integer*6


;*****************************
request elemcommuns
;*****************************

  access elemcommuns link code_etude of elemcommuns &
                     to code_etude of etudes optional
     item gt_code_etude = code_etude of elemcommuns
     item gt_date_debut_etude = date_debut_etude of elemcommuns
     item gt_type_aliment = type_aliment of elemcommuns
     item gt_race = race of elemcommuns
     item gt_espece = espece of elemcommuns
     item gt_etat = etat of elemcommuns
     item gt_voie_injection = voie_injection of elemcommuns
     item gt_type_echantillon = type_echantillon of elemcommuns
     item gt_etat_alimentaire = etat_alimentaire of elemcommuns
     item gt_vehicule_utilise = vehicule_utilise of elemcommuns

  output etudes alias etudesbis add if not record etudes exists
     item code_etude = code_etude of elemcommuns
     item date_debut_etude = date_debut_etude of elemcommuns

;**********************
request pre_int_cond_exp
;**********************

  access elemcommuns link type_echantillon of elemcommuns to type_echantillon &
   of condexperiment
   select condexperiment if vehicule_utilise = gt_vehicule_utilise &
             and etat_alimentaire = gt_etat_alimentaire &
             and voie_injection = gt_voie_injection

  output pre_int_cond_exp add
    item num_cond_res of pre_int_cond_exp final num_cond_res of condexperiment
    item methode_analyse of pre_int_cond_exp final methode_analyse &
                                                   of condexperiment
    item vol_administre of pre_int_cond_exp final vol_administre &
                                                  of condexperiment

;*************************
request acces_elemvariables
;*************************

  access elemvariables &
         link num_animal of elemvariables to num_animal of animal optional &
```

```
      and  num_animal of elemvariables to num_animal of pretest optional &
      and  num_animal of elemvariables to num_animal of test optional &
      and vol_administre_char of elemvariables to vol_administre_char &
                                          of pre_int_cond_exp optional

  output animal alias animalbis  add  if not record animal exists
     item num_animal final num_animal of elemvariables
     item num_fournisseur  final num_fournisseur of elemvariables
     item sexe  final sexe of elemvariables
     item espece  final gt_espece
     item race  final gt_race


  output pretest add update if not record pretest exists &
                         and (gt_etat = "PRETEST" or gt_etat = "TEST")
     item num_animal of pretest final num_animal of elemvariables
     item date_entree of pretest final date_entree of elemvariables
     item duree of pretest final duree of elemvariables
     item age_animal of pretest final age_animal of elemvariables
     item per_acclimatation of pretest final per_acclimatation of elemvariables

  output test add update if not record test exists &
                       and gt_etat = "TEST"
     item num_animal of test final num_animal of elemvariables
     item type_aliment of test final gt_type_aliment


  output INT_COND_EXP add if record pre_int_cond_exp exists
     item num_cond_res of INT_COND_EXP final num_cond_res &
                                 of pre_int_cond_exp
     item methode_analyse of INT_COND_EXP final methode_analyse &
                                          of pre_int_cond_exp
     item num_animal of INT_COND_EXP final num_animal of elemvariables



;***************
request num_max
;***************

   access num_max
      item gt_num_cond_res initial num_cond_res of num_max
      item gt_num_fournisseur final num_fournisseur of num_max
   subfile ancien_num_max keep include num_cond_res of num_max


;***************************
request elemhypervariables
;***************************

   access elemvariables LINK num_animal of elemvariables &
                        TO num_animal of elemhypervariables &
      link( methode_analyse of elemhypervariables + num_animal &
            of elemhypervariables) to methode_animal &
            of int_cond_exp optional &
      link  num_animal of elemhypervariables + nom_param_bio &
            of elemhypervariables + num_cond_res_char of int_cond_exp &
               + date_res_char of elemhypervariables &
                 to cle_tgv of resultat optional
```

```
sort on methode_analyse of elemhypervariables &
     on vol_administre of elemvariables &
     on num_animal

define nouveau char*25 = methode_analyse of elemhypervariables
define vol_administre_nouveau char*8 = vol_administre_char of elemvariables

temporary ancien char*25
temporary vol_administre_ancien char*8

temporary num_cond_res_precedent integer*10
    item  num_cond_res_precedent = gt_num_cond_res
item gt_num_cond_res = (gt_num_cond_res + 1) &
          if not record int_cond_exp exists and (ancien <> nouveau &
                        or vol_administre_ancien <> vol_administre_nouveau)

item ancien = nouveau
item vol_administre_ancien = vol_administre_nouveau

output condexperiment add  if not record int_cond_exp exists &
        and gt_num_cond_res <> num_cond_res_precedent
   item num_cond_res final gt_num_cond_res
   item voie_injection final gt_voie_injection
   item vol_administre final vol_administre of elemvariables
   item type_echantillon final gt_type_echantillon
   item methode_analyse final methode_analyse of elemhypervariables
   item vehicule_utilise final gt_vehicule_utilise
   item etat_alimentaire final gt_etat_alimentaire

output resultat alias resultatter add if not record int_cond_exp exists
   item date_res final date_res of elemhypervariables
   item num_cond_res final gt_num_cond_res
   item valeur_res final valeur_res of elemhypervariables
   item etat final gt_etat
   item commentaire final commentaire of elemhypervariables
   item nom_param_bio final nom_param_bio of elemhypervariables
   item num_animal final num_animal of elemhypervariables


output resultat alias resultatbis add  &
       if record int_cond_exp exists &
       and not record resultat exists &
       and (valeur_res of resultat <> valeur_res of elemhypervariables)

   item date_res final date_res of elemhypervariables
   item num_cond_res final num_cond_res of int_cond_exp
   item valeur_res final valeur_res of elemhypervariables
   item etat final gt_etat
   item commentaire final commentaire of elemhypervariables
   item nom_param_bio final nom_param_bio of elemhypervariables
   item num_animal final num_animal of elemhypervariables



;*****************************
request MISE_A_JOUR
;*****************************

access *ANCIEN_NUM_MAX link num_cond_res of ancien_num_max to &
                      num_cond_res of num_max
```

```
output num_max update if num_cond_res of num_max <> gt_num_cond_res
     item num_cond_res of num_max final gt_num_cond_res
     item num_fournisseur of num_max final gt_num_fournisseur


;*****************************
request EFFACAGE
;*****************************

access int_cond_exp
output int_cond_exp delete

;*****************************
request EFFACAGE_BIS
;*****************************

access pre_int_cond_exp
output pre_int_cond_exp delete

;*****************************
request maj_cl_numero
;*****************************

access tab_changements LINK nr_lot of tab_changements &
                         TO nr_lot of cl_numero optional

output cl_numero alias cl_numero_bis add if not record cl_numero exists
   item nr_lot final nr_lot of tab_changements
   item nr_def final nr_def of tab_changements

build
```

```
request fich_converti

  access fich_converti
  define gt_code_etude char*8 = parm prompt "Code de l'étude :" upshift

  sort on nr_animal

  define num_definitif char*16 = nr_animal + gt_code_etude
  temporary cptr integer
       item cptr initial 0
       item cptr = cptr + 1 if cptr < 2

  output elemhypervariables add
   item num_animal of elemhypervariables final num_definitif
   item nom_det of elemhypervariables final &
                     upshift (nom_det of fich_converti)
   item nom_param_bio of elemhypervariables final nom_param_bio &
                                                 of fich_converti
   item jour_etude of elemhypervariables final jour_etude of fich_converti
   item valeur_res of elemhypervariables final valeur_res of fich_converti


  output elemvariables add at start of nr_animal of fich_converti
    item num_animal of elemvariables final num_definitif
    item nr_animal of elemvariables final nr_animal of fich_converti
    item sexe of elemvariables final sexe of fich_converti

  output elemcommuns add if cptr = 1
    item code_etude of elemcommuns final gt_code_etude
    item etat of elemcommuns final "QUAR"

build transfert_ani
```

```
run transfert_pretest

;*********************************
   request fich_converti_pretest
;*********************************

     access fich_converti
     define gt_code_etude char*8 = parm prompt "Code de l'étude :" upshift
      sort on nr_animal

        define num_definitif char*16 = nr_animal + gt_code_etude
        temporary cptr integer
            item cptr initial 0
            item cptr = cptr + 1 if cptr < 2

     output elemhypervariables add if jour_etude of fich_converti < 0

       item num_animal of elemhypervariables final num_definitif
       item nom_det of elemhypervariables final &
               upshift(nom_det of fich_converti)
       item nom_param_bio of elemhypervariables final nom_param_bio &
                                                    of fich_converti
       item jour_etude of elemhypervariables final jour_etude of fich_converti
       item valeur_res of elemhypervariables final valeur_res of fich_converti


     output elemvariables add at start of nr_animal of fich_converti
       item num_animal of elemvariables final num_definitif
       item nr_animal of elemvariables final nr_animal of fich_converti
       item sexe of elemvariables final sexe of fich_converti

     output elemcommuns add if cptr = 1
       item code_etude of elemcommuns final gt_code_etude
       item etat of elemcommuns final "PRETEST"


;*************************************
request purge
;*************************************
; ( pour eliminer les elements de elemvariables qu'on aurait créés en trop
;   et auxquels ne sont rattachés aucuns résultats de la phase prétest )


  access elemvariables link num_animal of elemvariables to num_animal &
                                          of elemhypervariables optional

  output elemvariables DELETE if not record elemhypervariables exists



build
```

```
run transfert_test

;*********************************
   request fich_converti_test
;*********************************

     access fich_converti
     define gt_code_etude char*8 = parm prompt "Code de l'étude :" upshift
      sort on nr_animal

        define num_definitif char*16 = nr_animal + gt_code_etude
        temporary cptr integer
            item cptr initial 0
            item cptr = cptr + 1 if cptr < 2

     output elemhypervariables add if jour_etude of fich_converti > 0

       item num_animal of elemhypervariables final num_definitif
       item nom_det of elemhypervariables final &
                          upshift(nom_det of fich_converti)
       item nom_param_bio of elemhypervariables final nom_param_bio &
                                                  of fich_converti
       item jour_etude of elemhypervariables final jour_etude of fich_converti
       item valeur_res of elemhypervariables final valeur_res of fich_converti


     output elemvariables add at start of nr_animal of fich_converti
       item num_animal of elemvariables final num_definitif
       item nr_animal of elemvariables final nr_animal of fich_converti
       item sexe of elemvariables final sexe of fich_converti

     output elemcommuns add if cptr = 1
       item code_etude of elemcommuns final gt_code_etude
       item etat of elemcommuns final "TEST"


;*************************************
request purge
;*************************************
; ( pour eliminer les elements de elemvariables qu'on aurait créés en trop
;   et auxquels ne sont rattachés aucuns résultats de la phase prétest )


  access elemvariables link num_animal of elemvariables to num_animal &
                                          of elemhypervariables optional

  output elemvariables DELETE if not record elemhypervariables exists



build
```

```
run  maj_numanimal_pretest

global temporary gt_code_etude char*8

;-------------------------------------------------------------------------
    request acces_code_etude

      access elemcommuns
         item gt_code_etude = code_etude of elemcommuns

;-------------------------------------------------------------------------
    request maj_num_animal

      access elemvariables  LINK num_animal of elemvariables &
                             TO num_animal of elemhypervariables

      sorted on num_animal of elemvariables

      output elemhypervariables update
        item num_animal  final (nr_animal of elemvariables + gt_code_etude)

      output elemvariables update at num_animal of elemvariables
        item num_animal  final (nr_animal of elemvariables + gt_code_etude)


;-------------------------------------------------------------------------
build
```

```
request modification_num_animal

     access tab_changements  LINK (nr_def of tab_changements+ code_def of &
                                   tab_changements)&
                              TO num_animal of elemvariables &
                        AND   (nr_def of tab_changements + code_def of &
                                   tab_changements) &
                              TO num_animal of elemhypervariables

       define tri char*8 = (nr_lot of tab_changements + nr_def &
                           of tab_changements)
       sort  on  tri

     output elemhypervariables update if record elemhypervariables exists
         item num_animal final (nr_lot of tab_changements + code_def &
                                of tab_changements)
     output elemvariables update at start of tri &
                                 if record elemvariables exists
         item num_animal final (nr_lot of tab_changements + code_def &
                                of tab_changements)

build maj_numanimal_test
```

```
request QTP_1

access requete_biol &
       link date_entree of requete_biol to date_entree of pretest &
       link num_animal of pretest to num_animal of animal

output fruit_selection_1 add
       item num_fournisseur final num_fournisseur of animal
       item num_animal final num_animal of animal
       item espece final espece of animal
       item race final race of animal
       item sexe final sexe of animal

build QTP_1
```

```
un qtp_1_moins

lobal temporary date_entree_op date

*********************************
equest lecture_requete_biol
*********************************
 access requete_biol
    item date_entree_op = date_entree of requete_biol

************************************************
equest QTP_1_oper
************************************************
 access pretest LINK num_animal of pretest &
                TO num_animal of animal
   SELECT pretest IF date_entree of pretest < date_entree_op
(La LINK ne se fait que sur les PRETEST satisfaisant la condition)

 output fruit_selection_1 add
      item num_fournisseur final num_fournisseur of animal
      item num_animal final num_animal of animal
      item espece final espece of animal
      item race final race of animal
      item sexe final sexe of animal

uild
```

```
in qtp_1_plus

lobal temporary date_entree_op date

*********************************
equest lecture_requete_biol
*********************************
access requete_biol
   item date_entree_op = date_entree of requete_biol

*************************************************
equest QTP_1_oper
*************************************************
access pretest LINK num_animal of pretest &
               TO num_animal of animal
  SELECT pretest IF date_entree of pretest > date_entree_op
(La LINK ne se fait que sur les PRETEST satisfaisant la condition)

output fruit_selection_1 add
     item num_fournisseur final num_fournisseur of animal
     item num_animal final num_animal of animal
     item espece final espece of animal
     item race final race of animal
     item sexe final sexe of animal

uild
```

```
un QTP_2

lobal temporary gt_min_race char*20
lobal temporary gt_max_race char*20
lobal temporary gt_min_espece char*20
lobal temporary gt_max_espece char*20
lobal temporary gt_min_sexe char*1
lobal temporary gt_max_sexe char*1


*************************************
equest requete_biol
*************************************
ccess requete_biol
   item gt_min_race = race of requete_biol if race of requete_biol <> " "
   item gt_max_race = race of requete_biol if race of requete_biol <> " "
   item gt_min_race = "                    " if race of requete_biol = " "
   item gt_max_race = "~~~~~~~~~~~~~~~~~~~~" if race of requete_biol = " "

   item gt_min_espece = espece of requete_biol if espece of requete_biol <> " "
   item gt_max_espece = espece of requete_biol if espece of requete_biol <> " "
   item gt_min_espece = "                    " if espece of requete_biol = " "
   item gt_max_espece = "~~~~~~~~~~~~~~~~~~~~" if espece of requete_biol = " "

   item gt_min_sexe = sexe of requete_biol if sexe of requete_biol <> " "
   item gt_max_sexe = sexe of requete_biol if sexe of requete_biol <> " "
   item gt_min_sexe = "                    " if sexe of requete_biol = " "
   item gt_max_sexe = "~~~~~~~~~~~~~~~~~~~~" if sexe of requete_biol = " "

*************************************
equest QTP_2
*************************************
ccess animal
elect animal if (race of animal GE gt_min_race and &
                 race of animal LE gt_max_race) &
        and  (espece of animal GE gt_min_espece and &
                 espece of animal LE gt_max_espece) &
        and  (sexe of animal GE gt_min_sexe and &
                 sexe of animal LE gt_max_sexe)

utput fruit_selection_2 add
   item num_animal final num_animal of animal
   item num_fournisseur final num_fournisseur of animal


uild
```

```
un QTP_2_1

lobal temporary gt_min_race char*20
lobal temporary gt_max_race char*20
lobal temporary gt_min_espece char*20
lobal temporary gt_max_espece char*20
lobal temporary gt_min_sexe char*1
lobal temporary gt_max_sexe char*1


************************************
equest requete_biol
************************************
ccess requete_biol
   item gt_min_race = race of requete_biol if race of requete_biol <> " "
   item gt_max_race = race of requete_biol if race of requete_biol <> " "
   item gt_min_race = "                    " if race of requete_biol = " "
   item gt_max_race = "~~~~~~~~~~~~~~~~~~~~" if race of requete_biol = " "

   item gt_min_espece = espece of requete_biol if espece of requete_biol <> " "
   item gt_max_espece = espece of requete_biol if espece of requete_biol <> " "
   item gt_min_espece = "                    " if espece of requete_biol = " "
   item gt_max_espece = "~~~~~~~~~~~~~~~~~~~~" if espece of requete_biol = " "

   item gt_min_sexe = sexe of requete_biol if sexe of requete_biol <> " "
   item gt_max_sexe = sexe of requete_biol if sexe of requete_biol <> " "
   item gt_min_sexe = "                    " if sexe of requete_biol = " "
   item gt_max_sexe = "~~~~~~~~~~~~~~~~~~~~" if sexe of requete_biol = " "

************************************
equest QTP_2_1
************************************
ccess fruit_selection_1
elect fruit_selection_1 if (race of fruit_selection_1 GE gt_min_race and &
                race of fruit_selection_1 LE gt_max_race) &
        and  (espece of fruit_selection_1 GE gt_min_espece and &
                espece of fruit_selection_1 LE gt_max_espece) &
        and  (sexe of fruit_selection_1 GE gt_min_sexe and &
                sexe of fruit_selection_1 LE gt_max_sexe)

utput fruit_selection_2 add
   item num_animal final num_animal of fruit_selection_1
   item num_fournisseur final num_fournisseur of fruit_selection_1


uild
```

```
**************************************
equest QTP_3
**************************************
ccess REQUETE_BIOL &
      link nom_fournisseur of requete_biol to nom_fournisseur of fournisseur &
      link num_fournisseur of fournisseur to num_fournisseur of animal


utput fruit_selection_3 add
   item num_animal final num_animal of animal


uild QTP_3
```

```
**************************************
equest QTP_3_1
**************************************
ccess REQUETE_BIOL &
      link nom_fournisseur of requete_biol to nom_fournisseur of fournisseur &
      link num_fournisseur of fournisseur to num_fournisseur &
           of fruit_selection_1


utput fruit_selection_3 add
    item num_animal final num_animal of fruit_selection_1


uild QTP_3_1
```

```
**********************************
quest QTP_3_2
**********************************
cess REQUETE_BIOL &
    link nom_fournisseur of requete_biol to nom_fournisseur of fournisseur &
    link num_fournisseur of fournisseur to num_fournisseur &
         of fruit_selection_2


tput fruit_selection_3 add
  item num_animal final num_animal of fruit_selection_2


ild QTP_3_2
```

```
un QTP_4

lobal temporary gt_nom_param_bio char*25

************************************
equest requete_biol
************************************
ccess requete_biol
   item gt_nom_param_bio = nom_param_bio of requete_biol

************************************
equest QTP_4
************************************
ccess resultat
elect resultat if nom_param_bio of resultat = gt_nom_param_bio

utput fruit_selection_4 add
   item num_animal final num_animal of resultat
   item date_res final date_res of resultat
   item num_cond_res final num_cond_res of resultat
   item valeur_res final valeur_res of resultat
   item etat final etat of resultat


uild
```

```
in QTP_4_1

lobal temporary gt_nom_param_bio char*25

*************************************
equest requete_biol
*************************************
ccess requete_biol
   item gt_nom_param_bio = nom_param_bio of requete_biol

*************************************
equest QTP_4_1
*************************************
ccess fruit_selection_1 &
     link (num_animal of fruit_selection_1 + gt_nom_param_bio) &
          to cle_selection of resultat

itput fruit_selection_4 add
   item num_animal final num_animal of resultat
   item date_res final date_res of resultat
   item num_cond_res final num_cond_res of resultat
   item valeur_res final valeur_res of resultat
   item etat final etat of resultat


uild
```

```
an QTP_4_1_fin

lobal temporary gt_nom_param_bio char*25

************************************
equest requete_biol
************************************
ccess requete_biol
   item gt_nom_param_bio = nom_param_bio of requete_biol

************************************
equest QTP_4_1_fin
************************************
ccess fruit_selection_1 &
     link (num_animal of fruit_selection_1 + gt_nom_param_bio) &
          to cle_selection of resultat

utput fruit_selection_6 add
   item valeur_res final valeur_res of resultat
   item etat final etat of resultat
   item num_animal final num_animal of resultat
   item num_cond_res final num_cond_res of resultat
   item date_res final date_res of resultat

uild
```

```
un QTP_4_2

lobal temporary gt_nom_param_bio char*25

**********************************
equest requete_biol
**********************************
ccess requete_biol
   item gt_nom_param_bio = nom_param_bio of requete_biol

**********************************
equest QTP_4_2
**********************************
ccess fruit_selection_2 &
     link (num_animal of fruit_selection_2 + gt_nom_param_bio) &
          to cle_selection of resultat

utput fruit_selection_4 add
   item num_animal final num_animal of resultat
   item date_res final date_res of resultat
   item num_cond_res final num_cond_res of resultat
   item valeur_res final valeur_res of resultat
   item etat final etat of resultat


uild
```

```
un QTP_4_2_fin

lobal temporary gt_nom_param_bio char*25

************************************
equest requete_biol
************************************
ccess requete_biol
   item gt_nom_param_bio = nom_param_bio of requete_biol

************************************
equest QTP_4_2_fin
************************************
ccess fruit_selection_2 &
     link (num_animal of fruit_selection_2 + gt_nom_param_bio) &
          to cle_selection of resultat

utput fruit_selection_6 add
   item valeur_res final valeur_res of resultat
   item etat final etat of resultat
   item num_animal final num_animal of resultat
   item num_cond_res final num_cond_res of resultat
   item date_res final date_res of resultat

uild
```

```
un QTP_4_3

lobal temporary gt_nom_param_bio char*25

************************************
equest requete_biol
************************************
ccess requete_biol
   item gt_nom_param_bio = nom_param_bio of requete_biol

************************************
equest QTP_4_3
************************************
ccess fruit_selection_3 &
     link (num_animal of fruit_selection_3 + gt_nom_param_bio) &
          to cle_selection of resultat

utput fruit_selection_4 add
   item num_animal final num_animal of resultat
   item date_res final date_res of resultat
   item num_cond_res final num_cond_res of resultat
   item valeur_res final valeur_res of resultat
   item etat final etat of resultat


uild
```

```
un QTP_4_3_fin

lobal temporary gt_nom_param_bio char*25

************************************
equest requete_biol
************************************
ccess requete_biol
   item gt_nom_param_bio = nom_param_bio of requete_biol

************************************
equest QTP_4_3_fin
************************************
ccess fruit_selection_3 &
     link (num_animal of fruit_selection_3 + gt_nom_param_bio) &
          to cle_selection of resultat

utput fruit_selection_6 add
   item valeur_res final valeur_res of resultat
   item etat final etat of resultat
   item num_animal final num_animal of resultat
   item num_cond_res final num_cond_res of resultat
   item date_res final date_res of resultat

uild
```

```
n QTP_4_fin

obal temporary gt_nom_param_bio char*25

**********************************
quest requete_biol
**********************************
:cess requete_biol
   item gt_nom_param_bio = nom_param_bio of requete_biol

**********************************
:quest QTP_4_fin
**********************************
:cess resultat
:lect resultat if nom_param_bio of resultat = gt_nom_param_bio

itput fruit_selection_6 add
   item valeur_res final valeur_res of resultat
   item etat final etat of resultat
   item num_animal final num_animal of resultat
   item num_cond_res final num_cond_res of resultat
   item date_res final date_res of resultat

ild
```

```
n QTP_5_4

obal temporary gt_min_methode char*20
obal temporary gt_max_methode char*20
obal temporary gt_min_etat char*20
obal temporary gt_max_etat char*20


**********************************
quest requete_biol
**********************************
cess requete_biol
  item gt_min_methode = methode_analyse of requete_biol if methode_analyse &
                                of requete_biol <> " "
  item gt_max_methode = methode_analyse of requete_biol if methode_analyse &
                                of requete_biol <> " "
  item gt_min_methode = "                          " if methode_analyse &
                                of requete_biol = " "
  item gt_max_methode = "~~~~~~~~~~~~~~~~~~~~" if methode_analyse &
                                of requete_biol = " "

  item gt_min_etat = etat_alimentaire of requete_biol if etat_alimentaire &
                                of requete_biol <> " "
  item gt_max_etat = etat_alimentaire of requete_biol if etat_alimentaire &
                                of requete_biol <> " "
  item gt_min_etat = "                          " if etat_alimentaire &
                                of requete_biol = " "
  item gt_max_etat = "~~~~~~~~~~~~~~~~~~~~" if etat_alimentaire &
                                of requete_biol = " "


**********************************
equest QTP_5_4
**********************************
cess requete_biol &
      link type_echantillon to type_echantillon of condexperiment

elect condexperiment if (methode_analyse of condexperiment GE gt_min_methode &
      and  methode_analyse of condexperiment LE gt_max_methode) &
      and  (etat_alimentaire of condexperiment GE gt_min_etat &
           and etat_alimentaire of condexperiment LE gt_max_etat)


utput fruit_pre_selection_5 add
   item num_cond_res final num_cond_res of condexperiment

**********************************
equest PRE_SELECTION
**********************************
ccess fruit_pre_selection_5 &
       link num_cond_res to num_cond_res of fruit_selection_4

utput fruit_selection_5 add
  item num_animal final num_animal of fruit_selection_4
  item date_res final date_res of fruit_selection_4
  item valeur_res final valeur_res of fruit_selection_4
  item etat final etat of fruit_selection_4
  item num_cond_res final num_cond_res of fruit_selection_4

uild
```

```
n QTP_5_4_fin

obal temporary gt_min_methode char*20
obal temporary gt_max_methode char*20
obal temporary gt_min_etat char*20
obal temporary gt_max_etat char*20


**********************************
quest requete_biol
**********************************
cess requete_biol
  item gt_min_methode = methode_analyse of requete_biol if methode_analyse &
                        of requete_biol <> " "
  item gt_max_methode = methode_analyse of requete_biol if methode_analyse &
                        of requete_biol <> " "
  item gt_min_methode = "                    " if methode_analyse &
                        of requete_biol = " "
  item gt_max_methode = "~~~~~~~~~~~~~~~~~~~~" if methode_analyse &
                        of requete_biol = " "

  item gt_min_etat = etat_alimentaire of requete_biol if etat_alimentaire &
                        of requete_biol <> " "
  item gt_max_etat = etat_alimentaire of requete_biol if etat_alimentaire &
                        of requete_biol <> " "
  item gt_min_etat = "                    " if etat_alimentaire &
                        of requete_biol = " "
  item gt_max_etat = "~~~~~~~~~~~~~~~~~~~~" if etat_alimentaire &
                        of requete_biol = " "


**********************************
equest QTP_5_4_fin
**********************************
cess requete_biol &
      link type_echantillon to type_echantillon of condexperiment

elect condexperiment if (methode_analyse of condexperiment GE gt_min_methode &
      and  methode_analyse of condexperiment LE gt_max_methode) &
      and  (etat_alimentaire of condexperiment GE gt_min_etat &
           and etat_alimentaire of condexperiment LE gt_max_etat)


utput fruit_pre_selection_5 add
   item num_cond_res final num_cond_res of condexperiment

**********************************
equest PRE_SELECTION
**********************************
ccess fruit_pre_selection_5 &
       link num_cond_res to num_cond_res of fruit_selection_4

utput fruit_selection_6 add
  item valeur_res final valeur_res of fruit_selection_4
  item etat final etat of fruit_selection_4
  item num_cond_res final num_cond_res of fruit_selection_4
  item num_animal final num_animal of fruit_selection_4
  item date_res final date_res of fruit_selection_4

uild
```

```
in QTP_6_4

.obal temporary gt_age_animal integer
.obal temporary gt_operateur char*1

:*************************************
equest requete_biol
:*************************************
:cess requete_biol
   item gt_age_animal = age_animal of requete_biol
   item gt_operateur = operateur_age of requete_biol

:*************************************
equest QTP_6_4
:*************************************
:cess fruit_selection_4 &
     link num_animal of fruit_selection_4 to num_animal of pretest

efine age_reel integer = (days(date_res of fruit_selection_4) &
                          - days(date_entree of pretest) &
                          + age_animal of pretest)

utput fruit_selection_6 add &
         IF (age_reel = gt_age_animal and gt_operateur = "=" )
   item valeur_res final valeur_res of fruit_selection_4
   item etat final etat of fruit_selection_4
   item num_animal final num_animal of fruit_selection_4
   item num_cond_res final num_cond_res of fruit_selection_4
   item date_res final date_res of fruit_selection_4
utput fruit_selection_6 alias fruit_selection_moins add &
         IF (age_reel < gt_age_animal and gt_operateur = "<" )
   item valeur_res final valeur_res of fruit_selection_4
   item etat final etat of fruit_selection_4
   item num_animal final num_animal of fruit_selection_4
   item num_cond_res final num_cond_res of fruit_selection_4
   item date_res final date_res of fruit_selection_4
utput fruit_selection_6 alias fruit_selection_plus add &
         IF (age_reel > gt_age_animal and gt_operateur = ">" )
   item valeur_res final valeur_res of fruit_selection_4
   item etat final etat of fruit_selection_4
   item num_animal final num_animal of fruit_selection_4
   item num_cond_res final num_cond_res of fruit_selection_4
   item date_res final date_res of fruit_selection_4


uild
```

```
n QTP_6_5

obal temporary gt_age_animal integer
obal temporary gt_operateur char*1

**********************************
quest requete_biol
**********************************
cess requete_biol
   item gt_age_animal = age_animal of requete_biol
   item gt_operateur = operateur_age of requete_biol

**********************************
quest QTP_6_5
**********************************
cess fruit_selection_5 &
     link num_animal of fruit_selection_5 to num_animal of pretest

fine age_reel integer = (days(date_res of fruit_selection_5) &
                         - days(date_entree of pretest) &
                         + age_animal of pretest)

tput fruit_selection_6 add &
       IF (age_reel = gt_age_animal and gt_operateur = "=")
   item valeur_res final valeur_res of fruit_selection_5
   item etat final etat of fruit_selection_5
   item num_cond_res final num_cond_res of fruit_selection_5
   item num_animal final num_animal of fruit_selection_5
   item date_res final date_res of fruit_selection_5
tput fruit_selection_6 alias fruit_selection_6_moins add &
       IF (age_reel < gt_age_animal and gt_operateur = "<")
   item valeur_res final valeur_res of fruit_selection_5
   item etat final etat of fruit_selection_5
   item num_cond_res final num_cond_res of fruit_selection_5
   item num_animal final num_animal of fruit_selection_5
   item date_res final date_res of fruit_selection_5
tput fruit_selection_6 alias fruit_selection_6_plus add &
       IF (age_reel > gt_age_animal and gt_operateur = ">")
   item valeur_res final valeur_res of fruit_selection_5
   item etat final etat of fruit_selection_5
   item num_cond_res final num_cond_res of fruit_selection_5
   item num_animal final num_animal of fruit_selection_5
   item date_res final date_res of fruit_selection_5


uild
```

RAPPORTS (QUIZ)

```
ccess fruit_selection_6 link num_animal of fruit_selection_6 &
           to num_animal of pretest

efine age_reel integer = (days(date_res of fruit_selection_6) - &
                          days(date_entree of pretest) + age_animal of pretest)

(L'age reel est exprime en nombre de jours !)

eport age_reel &
      tab 10 &
      valeur_res of fruit_selection_6

et nohead
et report device disc
et report name AGE_PARAMETRE

uild traitement_age
```

```
access fruit_selection_6 LINK num_animal of fruit_selection_6 &
                         To num_animal of poids

  SELECT IF date_res of fruit_selection_6 = date_mesure of poids

report valeur_poids &
     tab 10 &
     valeur_res of fruit_selection_6

set nohead
set report device disc
set report name POIDS_PARAMETRE

uild traitement_poids
```

```
access fruit_selection_6 LINK num_animal of fruit_selection_6 &
                         To num_animal of animal

sort on sexe of animal

report sexe of animal &
     tab 2 &
     valeur_res of fruit_selection_6

set nohead
set report device disc
set report name SEXE_PARAMETRE

build traitement_sexe
```

```
access fruit_selection_6 LINK num_animal of fruit_selection_6 &
                         To num_animal of test

sort on type_aliment of test

report type_aliment of test &
     tab 10 &
     valeur_res of fruit_selection_6

set nohead
set report device disc
set report name TYPEALIMENT_PARAMETRE

uild traitement_typealiment
```

```
access fruit_selection_6 LINK num_animal of fruit_selection_6 &
                         To num_animal of animal &
                         LINK num_fournisseur of animal &
                         TO num_fournisseur of fournisseur

sort on nom_fournisseur of fournisseur

report nom_fournisseur of fournisseur &
     tab 25 &
     valeur_res of fruit_selection_6

set nohead
set report device disc
set report name FOURNISSEUR_PARAMETRE

uild traitement_fournisseur
```

```
access fruit_selection_6 LINK num_animal of fruit_selection_6 &
                         To num_animal of animal

sort on race of animal

report race of animal &
     tab 20 &
     valeur_res of fruit_selection_6

set nohead
set report device disc
set report name RACE_PARAMETRE

)uild traitement_race
```

```
cess fruit_selection_6 LINK num_animal of fruit_selection_6 &
                to num_animal of animal

rt on espece of animal

port espece of animal &
     tab 20 &
     valeur_res of fruit_selection_6

t nohead
t report device disc
t report name ESPECE_PARAMETRE

ild traitement_espece
```

```
ccess fruit_selection_6 link num_cond_res of fruit_selection_6 &
      to num_cond_res of condexperiment

ort on voie_injection of condexperiment

eport &
       voie_injection of condexperiment &
       tab 2 &
       valeur_res of fruit_selection_6

et nohead
et report device disc
et report name type_injection_parametre

uild traitement_type_injection
```

```
:cess requete_biol

:port &
  "Entry date      " &
  TAB 10 &
  date_entree &
  skip 1 &
  "Race            " &
  TAB 10 &
  race &
  skip 1 &
  "Species         " &
  TAB 10 &
  espece &
  skip 1 &
  "Sex             " &
  TAB 10 &
  sexe  &
  skip 1 &
  "Tradesman       " &
  TAB 10 &
  nom_fournisseur &
  skip 1 &
  "Parameter       " &
  TAB 10 &
  nom_param_bio &
  skip 1 &
  "Method          " &
  TAB 10 &
  methode_analyse &
  skip 1 &
  "Sample type     " &
  TAB 10 &
  type_echantillon &
  skip 1 &
  "State           " &
  TAB 10 &
  etat_alimentaire &
  skip 1 &
  "Age             " &
  TAB 10 &
  age_animal &
  skip 1 &
  "Stat type       " &
  TAB 10 &
  typestat &
  skip 1 &
  "Unit            " &
  TAB 10 &
  unite_mesure

et nohead
et report device disc
et report name INFORECHERCHE

uild prepa_inforecherche
```

```
ccess evopoidsetude link code_etude of evopoidsetude &
            to code_etude of etudes &
            link code_etude of evopoidsetude to code_etude of animal &
            link num_animal of animal to num_animal of poids

efine d_jour_etude integer = ( days(date_mesure of poids) - &
                        days(date_debut_etude of etudes ))

eport d_jour_etude &
      tab 10 &
      valeur_poids of poids

et nohead
et report device disc
et report name evopoids

uild evolution_poids
```

```
:cess evopoidsetude &
    link code_etude of evopoidsetude to code_etude of etudes &
and  code_etude of evopoidsetude to code_etude of animal &
    link num_animal of animal to num_animal of poids


    DEFINE d_jour_etude integer = ( days(date_mesure of poids) - &
                                    days(date_debut_etude of etudes) )

    DEFINE d_sexe_number integer = 1  IF sexe of animal = "M" &
                           ELSE   2  IF sexe of animal = "F"


 report  d_jour_etude &
         tab 5 &
         valeur_poids of poids &
         d_sexe_number

 set nohead
 set report device disc
 set report name EVOPOIDS_SEXE

uild evolution_poids_sexe
```

```
ccess EVOPOIDSAGE LINK espece of evopoidsage &
                  TO espece of ANIMAL

  DEFINE d_min_race char*20  = race of EVOPOIDSAGE IF race <> " "&
                         ELSE "                        " IF race = " "
  DEFINE d_max_race char*20  = race of EVOPOIDSAGE IF race <> " "&
                         ELSE "~~~~~~~~~~~~~~~~~~~~~~" IF race = " "
  DEFINE d_min_sexe char*20  = sexe of EVOPOIDSAGE IF sexe <> " "&
                         ELSE "                        " IF sexe = " "
  DEFINE d_max_sexe char*20  = sexe of EVOPOIDSAGE IF sexe <> " "&
                         ELSE "~" IF sexe = " "

 SELECT IF ((race of animal GE d_min_race and &
             race of animal LE d_max_race) AND &
            (sexe of animal GE d_min_sexe and &
             sexe of animal LE d_max_sexe))

eport num_animal of ANIMAL
et subfile name NUMEROS

ccess *NUMEROS LINK num_animal of *NUMEROS &
               TO num_animal of POIDS &
               LINK num_animal of POIDS &
               TO num_animal of PRETEST
  define age_reel integer = (days(date_mesure of poids) &
                             - days(date_entree of pretest) &
                             + age_animal of pretest)

EPORT age_reel &
      tab_5 &
      valeur_poids of poids

ET NOHEAD
ET REPORT DEVICE DISC
ET REPORT NAME  EVOPOIDSAGERES

uild evopoidsage
```

```
odule   organisation (input,output);

type

   string_10 = packed array[1..10] of char;
   string_20 = packed array[1..20] of char;
   string_6 = packed array[1..6] of char;
   string_25 = packed array[1..25] of char;


[global]  procedure orga_requete (var champ_1 : string_10 ;
                                  var champ_2 : string_20;
                                  var champ_3 : string_20;
                                  var champ_4 : char;
                                  var champ_5 : integer;
                                  var champ_6 : integer;
                                  var champ_7 : string_6;
                                  var champ_8 : string_20;
                                  var champ_9 : string_25;
                                  var champ_10 : string_25;
                                  var qtp_1_def : char;
                                  var qtp_2_def : char;
                                  var qtp_3_def : char;
                                  var qtp_4_def : char;
                                  var qtp_5_def : char;
                                  var qtp_6_def : char;
                                  var qtp_2_1_def : char;
                                  var qtp_3_1_def : char;
                                  var qtp_3_2_def : char;
                                  var qtp_4_1_def : char;
                                  var qtp_4_2_def : char;
                                  var qtp_4_3_def : char;
                                  var qtp_5_1_def : char;
                                  var qtp_5_2_def : char;
                                  var qtp_5_3_def : char;
                                  var qtp_5_4_def : char;
                                  var qtp_6_4_def : char;
                                  var qtp_6_5_def : char;
                                  var qtp_5_4_fin_def : char;
                                  var qtp_5_3_fin_def : char;
                                  var qtp_5_2_fin_def : char;
                                  var qtp_4_1_fin_def : char;
                                  var qtp_4_2_fin_def : char;
                                  var qtp_4_3_fin_def : char;
                                  var qtp_4_fin_def : char);


var
   tableau_selection : array[1..6] of integer;
   i : integer;
   qtp_1 : boolean;
   qtp_2 : boolean;
   qtp_3 : boolean;
   qtp_4 : boolean;
   qtp_5 : boolean;
   qtp_6 : boolean;
   qtp_2_1 : boolean;
   qtp_3_1 : boolean;
   qtp_3_2 : boolean;
   qtp_4_1 : boolean;
```

```
  qtp_4_2 : boolean;
  qtp_4_3 : boolean;
  qtp_5_1 : boolean;
  qtp_5_2 : boolean;
  qtp_5_3 : boolean;
  qtp_5_4 : boolean;
  qtp_6_4 : boolean;
  qtp_6_5 : boolean;
  qtp_5_4_fin : boolean;
  qtp_5_3_fin : boolean;
  qtp_5_2_fin : boolean;
  qtp_4_1_fin : boolean;
  qtp_4_2_fin : boolean;
  qtp_4_3_fin : boolean;
  qtp_4_fin : boolean;


begin
  for i:= 1 to 6 do
     tableau_selection [i] := 0 ;
  qtp_1 := false;
  qtp_2 := false;
  qtp_3 := false;
  qtp_4 := false;
  qtp_5 := false;
  qtp_6 := false;
  qtp_2_1 := false;
  qtp_3_1 := false;
  qtp_3_2 := false;
  qtp_4_1 := false;
  qtp_4_2 := false;
  qtp_4_3 := false;
  qtp_5_1 := false;
  qtp_5_2 := false;
  qtp_5_3 := false;
  qtp_5_4 := false;
  qtp_6_4 := false;
  qtp_6_5 := false;
  qtp_5_4_fin := false;
  qtp_5_3_fin := false;
  qtp_5_2_fin := false;
  qtp_4_1_fin := false;
  qtp_4_2_fin := false;
  qtp_4_3_fin := false;
  qtp_4_fin := false;


*  etablissement du tableau comportant les etapes de selection  *)

  i := 1;
  IF champ_6 <> 0
     then
        begin
          tableau_selection [i] := 1 ;
          i := i + 1
        end;
  IF (champ_2 <> ' ') or (champ_1 <> ' ') or (champ_4 <> ' ')
     then
        begin
          tableau_selection [i] := 2 ;
```

```
        i := i + 1
      end;
  IF champ_3 <> ' '
     then
      begin
        tableau_selection [i] := 3 ;
        i := i + 1
      end;
  IF champ_9 <> ' '
     then
      begin
        tableau_selection [i] := 4 ;
        i := i + 1
      end;
  IF (champ_10 <> ' ') or (champ_8 <> ' ') or (champ_7 <> ' ')
     then
      begin
        tableau_selection [i] := 5 ;
        i := i + 1
      end;
  IF champ_5 <> 0
     then
      begin
        tableau_selection [i] := 6 ;
        i := i + 1
      end;

* mise a jour des booleens pour les requetes a executer *)

  i := 1;
  if tableau_selection [i] <> 0
     then
       BEGIN
        case tableau_selection [i] of
          1 : qtp_1 := true;
          2 : qtp_2 := true;
          3 : qtp_3 := true;
          4 : qtp_4 := true;
          5 : qtp_5 := true;
          6 : qtp_6 := true
        end;
        i := 2;
        if tableau_selection [i] <> 0
           then
             begin
               case tableau_selection [i] of
                  2: qtp_2_1 := true;
                  3: case tableau_selection [i - 1] of
                       1: qtp_3_1 := true;
                       2: qtp_3_2 := true
                     end;
                  4: case tableau_selection [i - 1] of
                       1: qtp_4_1 := true;
                       2: qtp_4_2 := true;
                       3: qtp_4_3 := true
                     end;
                  5: case tableau_selection [i - 1] of
                       1: qtp_5_1 := true;
                       2: qtp_5_2 := true;
                       3: qtp_5_3 := true;
```

```
        4: qtp_5_4 := true
      end;
   6: case tableau_selection [i - 1] of
        4: qtp_6_4 := true;
        5: qtp_6_5 := true
      end;
end;
i := 3;
if tableau_selection [i] <> 0
   then
      BEGIN
        case tableau_selection [i] of
          3: qtp_3_2 := true;
          4: case tableau_selection [i - 1] of
               2: qtp_4_2 := true;
               3: qtp_4_3 := true
             end;
          5: case tableau_selection [i - 1] of
               2: qtp_5_2 := true;
               3: qtp_5_3 := true;
               4: qtp_5_4 := true
             end;
          6: case tableau_selection [i - 1] of
               4: qtp_6_4 := true;
               5: qtp_6_5 := true
             end;
        end;
        i := 4;
        if tableau_selection [i] <> 0
           then
             begin
               case tableau_selection [i] of
                 4: qtp_4_3 := true ;
                 5: case tableau_selection [i - 1] of
                      3: qtp_5_3 := true;
                      4: qtp_5_4 := true
                    end;
                 6: case tableau_selection [i - 1] of
                      4: qtp_6_4 := true;
                      5: qtp_6_5 := true
                    end;
               end;
               i := 5 ;
               if tableau_selection [i] <> 0
                  then
                    BEGIN
                      case tableau_selection [i] of
                        5: qtp_5_4 := true;
                        6: case tableau_selection [i - 1] of
                             4: qtp_6_4 := true;
                             5: qtp_6_5 := true
                           end;
                      end;
                      i := 6;
                      if tableau_selection [i] <> 0
                         then
                            qtp_6_5 := true
                         else
                           if tableau_selection [i - 1] = 5
                               then
```

```
                                    begin
                                     qtp_5_4_fin := true;
                                     qtp_5_4 := false
                                    end;
                        END
                      else
                        BEGIN
                            case tableau_selection [i - 1] of
                              4: begin
                                   qtp_4_3_fin := true;
                                   qtp_4_3 := false
                                 end;
                              5: case tableau_selection [i - 2] of
                                   3: begin
                                        qtp_5_3_fin := true;
                                        qtp_5_3 := false
                                      end;
                                   4: begin
                                        qtp_5_4_fin := true;
                                        qtp_5_4 := false
                                      end;
                                 end;
                            end;
                        END;
                  end
                else
                  begin
                    case tableau_selection [i - 1] of
                       4: case tableau_selection [i - 2] of
                            2: begin
                                 qtp_4_2_fin := true;
                                 qtp_4_2 := false
                               end;
                            3: begin
                                 qtp_4_3_fin := true;
                                 qtp_4_3 := false
                               end;
                          end;
                       5: case tableau_selection [i - 2] of
                            2: begin
                                 qtp_5_2_fin := true;
                                 qtp_5_2 := false
                               end;
                            3: begin
                                 qtp_5_3_fin := true;
                                 qtp_5_3 := false
                               end;
                            4: begin
                                 qtp_5_4_fin := true;
                                 qtp_5_4 := false
                               end;
                          end;
                    end;
                  end;
             END
          else
             BEGIN
               case tableau_selection [i - 1] of
                   4: case tableau_selection [i - 2] of
                        1: begin
```

```
                                    qtp_4_1_fin := true;
                                    qtp_4_1 := false
                                  end;
                               2: begin
                                    qtp_4_2_fin := true;
                                    qtp_4_2 := false
                                  end;
                               3: begin
                                    qtp_4_3_fin := true;
                                    qtp_4_3 := false
                                  end;
                            end;
                         5: begin
                              qtp_5_4_fin := true;
                              qtp_5_4 := false
                            end;
                      end;
                    END;
            end
          else
             if tableau_selection [i - 1] = 4
             then
                begin
                  qtp_4_fin := true;
                  qtp_4 := false
                end;

      END;

* garnissage des parametres passes avec la bonne valeur [T ou F]  *)

  if qtp_1
     then
        qtp_1_def := 'T'
     else
        qtp_1_def := 'F';
  if qtp_2
     then
        qtp_2_def := 'T'
     else
        qtp_2_def := 'F';
  if qtp_3
     then
        qtp_3_def := 'T'
     else
        qtp_3_def := 'F';
  if qtp_4
     then
        qtp_4_def := 'T'
     else
        qtp_4_def := 'F';
  if qtp_5
     then
        qtp_5_def := 'T'
     else
        qtp_5_def := 'F';
  if qtp_6
     then
        qtp_6_def := 'T'
     else
```

```
      qtp_6_def := 'F';
if qtp_2_1
   then
      qtp_2_1_def := 'T'
   else
      qtp_2_1_def := 'F';
if qtp_3_1
   then
      qtp_3_1_def := 'T'
   else
      qtp_3_1_def := 'F';
if qtp_3_2
   then
      qtp_3_2_def := 'T'
   else
      qtp_3_2_def := 'F';
if qtp_4_1
   then
      qtp_4_1_def := 'T'
   else
      qtp_4_1_def := 'F';
if qtp_4_2
   then
      qtp_4_2_def := 'T'
   else
      qtp_4_2_def := 'F';
if qtp_4_3
   then
      qtp_4_3_def := 'T'
   else
      qtp_4_3_def := 'F';
if qtp_5_1
   then
      qtp_5_1_def := 'T'
   else
      qtp_5_1_def := 'F';
if qtp_5_2
   then
      qtp_5_2_def := 'T'
   else
      qtp_5_2_def := 'F';
if qtp_5_3
   then
      qtp_5_3_def := 'T'
   else
      qtp_5_3_def := 'F';
if qtp_5_4
   then
      qtp_5_4_def := 'T'
   else
      qtp_5_4_def := 'F';
if qtp_6_4
   then
      qtp_6_4_def := 'T'
   else
      qtp_6_4_def := 'F';
if qtp_6_5
   then
      qtp_6_5_def := 'T'
   else
```

```
      qtp_6_5_def := 'F';

  if qtp_4_fin
     then
        qtp_4_fin_def := 'T'
     else
        qtp_4_fin_def := 'F';
  if qtp_4_3_fin
     then
        qtp_4_3_fin_def := 'T'
     else
        qtp_4_3_fin_def := 'F';
  if qtp_4_2_fin
     then
        qtp_4_2_fin_def := 'T'
     else
        qtp_4_2_fin_def := 'F';
  if qtp_4_1_fin
     then
        qtp_4_1_fin_def := 'T'
     else
        qtp_4_1_fin_def := 'F';
  if qtp_5_2_fin
     then
        qtp_5_2_fin_def := 'T'
     else
        qtp_5_2_fin_def := 'F';
  if qtp_5_3_fin
     then
        qtp_5_3_fin_def := 'T'
     else
        qtp_5_3_fin_def := 'F';
  if qtp_5_4_fin
     then
        qtp_5_4_fin_def := 'T'
     else
        qtp_5_4_fin_def := 'F';


end;

ND.
```

```
module conversion  (input,output) ;


global] procedure conv_xybion;


type
  enreg_converti = record
      nom_det : packed array [1..25] of char;
      nr_animal :  packed array [1..8] of char;
      sexe : char;
      dods : integer;
      nom_param :  packed array [1..25] of char;
      valeur : double
    end;
  nom_fichier = packed array [1..25] of char;

var
  reponse : char;
  reponse_determinant : char;
  continuer : boolean;
  i : integer;
  fichier_xybion : text;
  fich_converti : file of enreg_converti;
  filename :packed array [1..25] of char;
  line :  packed array [1..62] of char;
  ligne_recopiee : enreg_converti;
  determinant : packed array [1..25] of char;
  b1 :  packed array [1..2] of char;
  nom_param : packed array [1..4] of char;
  b2 : packed array [1..2] of char;
  nr_animal :packed array [1..8] of char;
  b3 : packed array [1..2] of char;
  sexe : char;
  b4 : packed array [1..2] of char;
  group : packed array [1..2] of char;
  b5 : packed array [1..2] of char;
  sous_group : char;
  b6 : packed array [1..2] of char;
  dods : packed array [1..4] of char;
  b7 : packed array [1..8] of char;
  resultat : packed array [1..8] of char;
  b8 : packed array [1..14] of char;

  demande_acceptee : boolean;
  fichierentre : array [1..20] of nom_fichier;
  j : integer;


procedure PROC_DODS;
var
   un : integer;
   dix: integer;
   cen: integer;
   mil: integer;
   signe_negatif : boolean;

begin
   signe_negatif := false;
   if dods[4] <> ' '
```

```
    then
      begin
        un := ord(dods[4]) - 48;
        ligne_recopiee.dods := un;
        if dods [3] <> ' '
          then
            begin
              if dods[3]='-'
                then
                  signe_negatif := true
                else
                  begin
                    dix := ord(dods[3]) - 48;
                    ligne_recopiee.dods := ligne_recopiee.dods  + (dix * 10);
                  end;
              if dods[2] <> ' '
                then
                  begin
                    if dods[2]='-'
                      then
                        signe_negatif := true
                      else
                        begin
                          cen := ord(dods[2]) - 48;
                          ligne_recopiee.dods := ligne_recopiee.dods
                                                 + (cen * 100);
                        end;
                    if dods [1] <> ' '
                      then
                        begin
                          if dods[1]='-'
                            then
                              signe_negatif := true
                            else
                              begin
                                mil := ord(dods[1]) - 48;
                                ligne_recopiee.dods := ligne_recopiee.dods
                                                       + (mil * 1000)
                              end;
                        end;
                  end;
            end;
      end;
 if signe_negatif
   then
     ligne_recopiee.dods := - (ligne_recopiee.dods)
nd;


rocedure PROC_VALEUR;
ar
 i: integer;
 position_point: integer;
 coefficient: double;
 bool : boolean;


rocedure DETERMINATION_VALEUR(borne:integer;facteur_coeff:double;
                              incre_decre :integer);
egin
```

```
bool := false;
while (i <> borne) and (not bool) do
  begin
    if resultat[i] <>' '
      then
        begin
          ligne_recopiee.valeur :=  ligne_recopiee.valeur +
                                    ((ord(resultat[i]) - 48) * coefficient);
          i:= i + incre_decre;
          coefficient := coefficient * facteur_coeff
        end
       else
        bool := true;
   end;
nd;

egin
 i:= 1;
 bool := false;
 while (i < 9) and ( not bool)  do
    begin
      if resultat[i] <> '.'
        then i := i+1
        else bool := true
    end;
 position_point := i;
 coefficient := 1;
 ligne_recopiee.valeur := 0;
 i := i-1;
 determination_valeur(0,10,-1);

 coefficient := 0.1;
 i := position_point;
 if i <> 9
   then i:= i+1;
 determination_valeur(9,0.1,1)
nd;

procedure verification ;

  var
    k : integer;
    fin : boolean;

  begin
    k := 1;
    fin := false;
    demande_acceptee := true;
    while (k < 21) and (not fin)  do
       begin
          if filename = fichierentre [k]
             then
               begin
                 writeln('Ce fichier a deja ete introduit!');
                 demande_acceptee := false;
                 fin := true;
               end
             else
               k := k + 1
       end;
```

```
 end;


GIN
 initialisation du tableau contenant les noms de fichiers *)
for j := 1 to 20 do
    fichierentre [j] := ' ';
j := 1;


write('Pour quel determinant : ');
readln (determinant);
write('Entrer le nom du fichier extraction : ');
read(filename);
fichierentre [j] := filename;
j := j + 1;
open(fich_converti,'fich_converti.dat');
rewrite(fich_converti);
open(fichier_xybion,filename,history := old);
reset (fichier_xybion);

while not eof(fichier_xybion) do
  begin
     readln(fichier_xybion,b1,nom_param,b2,nr_animal,b3,sexe,b4,group,
          b5,sous_group,b6,dods,b7,resultat,b8);
    if group = ' 1'
    then
        begin
          for i := 1 to 25 do
            ligne_recopiee.nom_det[i] := determinant[i];
           for i := 1 to 8 do
            ligne_recopiee.nr_animal[i] := nr_animal[i];
          ligne_recopiee.sexe := sexe;
           proc_dods;
           for i := 1 to 4 do
            ligne_recopiee.nom_param[i] := nom_param[i];
           for i:= 5 to 25 do
            ligne_recopiee.nom_param[i] := ' ';
          proc_valeur;
           write(fich_converti,ligne_recopiee)
        end;
  end;

close(fichier_xybion);
close(fich_converti);

*         demande d'autres parametres pour le meme determinant *)

EPEAT
 repeat
   writeln('Existe-t-il d''autres fichiers extractions ');
   writeln('pour ce meme determinant (Y/N) ?');
   readln (reponse)
  until (reponse = 'y') or (reponse = 'Y')
        or (reponse = 'n') or (reponse = 'N');

demande_acceptee := true;
repeat
 if (reponse = 'y') or (reponse = 'Y')
   then
```

```
    begin
      continuer := true;
      writeln('Entrer le nom du fichier : ');
      readln(filename);
      verification
    end
   else
      continuer := false
until demande_acceptee;

fichierentre [j] := filename;
j := j + 1;


while continuer do
  begin
    open(fich_converti,'FICH_CONVERTI.DAT',history:=old);
    extend(fich_converti);
    open(fichier_xybion,filename,history := old);
    reset (fichier_xybion);

    while not eof(fichier_xybion) do
      begin
        readln(fichier_xybion,b1,nom_param,b2,nr_animal,b3,sexe,b4,group,
           b5,sous_group,b6,dods,b7,resultat,b8);
        if group = ' 1'
          then
            begin
              for i := 1 to 25 do
                  ligne_recopiee.nom_det[i] := determinant[i];
              for i := 1 to 8 do
                  ligne_recopiee.nr_animal[i] := nr_animal[i];
              ligne_recopiee.sexe := sexe;
              proc_dods;
              for i := 1 to 4 do
                  ligne_recopiee.nom_param[i] := nom_param[i];
              for i:= 5 to 25 do
                  ligne_recopiee.nom_param[i] := ' ';
              proc_valeur;
              write(fich_converti,ligne_recopiee)
            end;
      end;
    close(fichier_xybion);
    close(fich_converti);

repeat
  writeln('Existe-t-il d''autres fichiers extractions ');
  writeln('pour ce meme determinant (Y/N) ?');
  readln (reponse)
 until (reponse = 'y') or (reponse = 'Y')
       or (reponse = 'n') or (reponse = 'N');

demande_acceptee := true;
repeat

if (reponse = 'y') or (reponse = 'Y')
  then
    begin
      continuer := true;
      writeln('Entrer le nom du fichier : ');
```

```
      readln(filename);
      verification
    end
   else
      continuer := false
 until demande_acceptee ;

 fichierentre [j] := filename;
 j := j + 1

end;

repeat
  writeln('Voulez vous continuer pour un autre déterminant [Y/N] ?');
  readln(reponse_determinant)
until (reponse_determinant = 'y') or (reponse_determinant = 'Y')
   or (reponse_determinant = 'n') or (reponse_determinant = 'N');

if (reponse_determinant = 'y') or (reponse_determinant = 'Y')
  then
    BEGIN
      writeln('Pour quel determinant : ');
      readln (determinant);
      writeln('Entrer le nom du fichier extraction : ');
      readln(filename);
      fichierentre [j] := filename;
      j := j + 1;
      open(fich_converti,'fich_converti.dat',history := old);
      extend(fich_converti);
      open(fichier_xybion,filename,history := old);
      reset (fichier_xybion);

      while not eof(fichier_xybion) do
        begin
          readln(fichier_xybion,b1,nom_param,b2,nr_animal,b3,sexe,b4,group,
                 b5,sous_group,b6,dods,b7,resultat,b8);
          if group = ' 1'
           then
             begin
               for i := 1 to 25 do
               ligne_recopiee.nom_det[i] := determinant[i];
               for i := 1 to 8 do
               ligne_recopiee.nr_animal[i] := nr_animal[i];
               ligne_recopiee.sexe := sexe;
               proc_dods;
               for i := 1 to 4 do
               ligne_recopiee.nom_param[i] := nom_param[i];
               for i:= 5 to 25 do
               ligne_recopiee.nom_param[i] := ' ';
               proc_valeur;
               write(fich_converti,ligne_recopiee)
             end;
        end;

      close(fichier_xybion);
      close(fich_converti);
    END;

NTIL (reponse_determinant = 'N') or (reponse_determinant = 'n');
```

d;

D.

STATISTIQUES (RS/1)

```
:ocedure;

:PONSE = gettext("Search from clinical biology (B) or toxicology (T) ?");
) WHILE (REPONSE <> "B") or (REPONSE <> "T");
  type "ERROR You must enter 'B' or 'T' !";
  REPONSE = gettext("Search from clinical biology (B) or toxicology (T) ?");
:ND;
: REPONSE = "B"
  then
:gin;
:ll $ez_readfile("inforecherche.txt","inforecherche",,,,,,true);
 = getcell("inforecherche",1,11);
: r = "TA"
 then
   begin;
     call $ez_readfile("age_parametre.txt","trtage",,,,,,true);
     call traitement_age;
   end;
lse
 if r = "TP"
   then
     begin;
       call $ez_readfile("poids_parametre.txt","trtpoids",,,,,,true);
       call traitement_poids;
     end;
lse
 if r = "TS"
   then
     begin;
       call $ez_readfile("sexe_parametre.txt","trtsexe",,,,,,true);
       set NOMTABLE = "trtsexe";
       set NOMGRAPHE = "trtsexegraph";
       set LIBELLE1 = "Do you want to test the difference between sexes ?";
       set LIBELLE2 = "There is a difference between the two sexes !";
       call traitement(NOMTABLE,NOMGRAPHE,LIBELLE1,LIBELLE2);
     end;
lse
 if r = "TAL"
   then
     begin;
       call $ez_readfile("typealiment_parametre.txt","trtali",,,,,,true);
       set NOMTABLE = "trtali";
       set NOMGRAPHE = "trtaligraph";
       set LIBELLE1 = "Do you want to test the aliment type effect ?";
       set LIBELLE2 = "There is a aliment type effect !";
       call traitement(NOMTABLE,NOMGRAPHE,LIBELLE1,LIBELLE2);
     end;
lse
 if r = "TF"
   then
     begin;
       call $ez_readfile("fournisseur_parametre.txt","trtfourn",,,,,,true);
       set NOMTABLE = "trtfourn";
       set NOMGRAPHE = "trtfourngraph";
       set LIBELLE1 = "Do you want to test the tradesman effect ?";
       set LIBELLE2 = "There is a tradesman effect !";
       call traitement(NOMTABLE,NOMGRAPHE,LIBELLE1,LIBELLE2);
     end;
lse
 if r = "TR"
```

```
  then
    begin;
      call $ez_readfile("race_parametre.txt","trtsouche",,,,,,true);
      set NOMTABLE = "trtsouche";
      set NOMGRAPHE = "trtsouchegraph";
      set LIBELLE1 = "Do you want to test the race effect ?";
      set LIBELLE2 = "There is a race effect !";
      call traitement(NOMTABLE,NOMGRAPHE,LIBELLE1,LIBELLE2);
    end;
se
if r = "TI"
  then
    begin;
      call $ez_readfile("type_injection_parametre.txt","trtinjection",,,,,
                        ,true);
      set NOMTABLE = "trtinjection";
      set NOMGRAPHE = "trtinjectiongraph";
      set LIBELLE1 = "Do you want to test the injection type effect ?";
      set LIBELLE2 = "There is a injection type effect !";
      call traitement(NOMTABLE,NOMGRAPHE,LIBELLE1,LIBELLE2);
    end;
lse
 if r = "ESP"
   then
     begin;
       call $ez_readfile("espece_parametre.txt","trtespece",,,,,,true);
       set NOMTABLE = "trtespece";
       set NOMGRAPHE = "trtespecegraph";
       set LIBELLE1 = "Do you want to test the species type effect ?";
       set LIBELLE2 = "There is a species type effect !";
       call traitement(NOMTABLE,NOMGRAPHE,LIBELLE1,LIBELLE2);
     end;
D;
/* Fin de ce qui concerne la biologie clinique ,c'est à dire le fichier
   INFORECHERCHE
*/

lse
 if REPONSE = "T"
    then
gin;
ll $ez_readfile("recherchetoxico.txt","recherchetoxico",,,,,,true);
= getcell("recherchetoxico",1,5);

 if r = "EPE"
   then
     begin;
       call $ez_readfile("evopoids.txt","evopoids",,,,,,true);
       call evo_poids;
     end;
else
  if r = "EPES"
    then
      begin;
        call $ez_readfile("evopoids_sexe.txt","evopetsexe",,,,,,true);
        call evo_poids_etd_sexe;
      end;
 else
   if r = "EPA"
     then
```

```
        begin;
          call $ez_readfile("evopoidsageres.txt","evopage",,,,,,true);
          call evo_poids_age;
        end;
 else
   if r = "EPR"
     then
       begin;
         call $ez_readfile("evopoidsagerace.txt","evoprace",,,,,,true);
         call evo_poids_race;
       end;
else
  if r = "EPS"
    then
      begin;
        call $ez_readfile("evopoidsagesexe.txt","evopsexe",,,,,,true);
        call evo_poids_sexe;
      end;
end;
nd;
```

```
OCEDURE(NORMALITE,MOYENNE,STD,TABLEVAL,UPLIMIT,LOWLIMIT,SEUIL,INDICE,
        SAMPLENAME);

QUEST = encode("Enter threshold (95 or 99)in % for limits for ","A(47)")
                 cat encode(SAMPLENAME,"A(15)");
QUESTION = encode(QUEST,"A(62)") cat encode(":","A(1)");

if NORMALITE = "NSIG"   /* pour les echantillons normaux */
   then
     begin;
      t = getnumber(QUESTION);
      DO WHILE (t <> 95) and (t <> 99);
         type "You must enter '99' or '95' !";
         t = getnumber(QUESTION);
      END;
      if t = 95
         then
            begin;
              LOWLIMIT = MOYENNE - ( 1.96 * STD);
              UPLIMIT = MOYENNE + ( 1.96 * STD);
              SEUIL = "95% " ;
            end;
         else
           if t = 99
            then
               begin;
                 LOWLIMIT = MOYENNE - ( 2.58 * STD);
                 UPLIMIT = MOYENNE + ( 2.58 * STD);
                 SEUIL = "99% " ;
               end;
     end;
   else
      if NORMALITE = "SIG"
         then
           begin;
             sort table(TABLEVAL) into TABLEVALBIS by col INDICE ;
             SAMPLESIZE = count of col INDICE of table(TABLEVAL);
             t = getnumber(QUESTION);
             DO WHILE (t <> 95) and (t <> 99);
               type "You must enter '99' or '95' !";
               t = getnumber(QUESTION);
             END;
             if t = 95
                then
                  begin;
                    COEFFICIENT1 = 0.025;
                    COEFFICIENT2 = 0.975;
                    SEUIL = "95% ";
                  end;
                else
                  begin;
                    COEFFICIENT1 = 0.005;
                    COEFFICIENT2 = 0.995;
                    SEUIL = "99% ";
                  end;


              RANGMIN = COEFFICIENT1 * (SAMPLESIZE + 1);
              RANGMAX = COEFFICIENT2 * (SAMPLESIZE + 1);
              typededonnees = type(RANGMIN);
```

```
        typededonnees2 = type(RANGMAX);
        if typededonnees = "FLOAT"
          then
            begin;
              if RANGMIN < 1
                 then
                   LOWLIMIT = (row 1 col INDICE of TABLEVALBIS);
                 else
                   begin;
                     VAL1 = ceil(RANGMIN) - 1;
                     VAL2 = ceil(RANGMIN);
                     LOWLIMIT = (row VAL1 col INDICE of TABLEVALBIS
                           + row VAL2 col INDICE of TABLEVALBIS)/2;
                   end;
            end;
          else
            begin;
             LOWLIMIT = row RANGMIN col INDICE of TABLEVALBIS;
            end;
        if typededonnees2 = "FLOAT"
          then
           begin;
             if RANGMAX > SAMPLESIZE
                then
                   UPLIMIT = (row SAMPLESIZE col INDICE of TABLEVALBIS);
                else
                  begin;
                    VAL1 = ceil(RANGMAX) - 1;
                    VAL2 = ceil(RANGMAX);
                    UPLIMIT = (row VAL1 col INDICE of TABLEVALBIS
                             + row VAL2 col INDICE of TABLEVALBIS)/2;
                  end;
           end;
          else
            begin;
              UPLIMIT = row RANGMAX col INDICE of TABLEVALBIS;
            end;
      delete TABLEVALBIS;

    end;
end;
```

```
ocedure;

make graph trtagegraph from col 0 of trtage vs col 1 of trtage;
type "Try to fit line...";
call $FITLINE("trtagegraph", 1, equ ,true, "anova",,,false);
R1 = getcell("anova",1,7);
Seuil = getnumber("Give the threshold you want for fit : ");
ajustement = "Alineaire";
if R1 <= seuil
   then
     begin;
       type "Try to fit polynomial (second degree) ...";
       call $fitpolynomial("trtagegraph",1,2,,true,"anovapoly2",,,false);
       R2 = getcell("anovapoly2",1,7);
       if R2 > R1
           then
             begin;
              delete curve 2 of trtagegraph;
              Rmax = R2;
              ajustement = "Apolynomial";
             end;
           else
             begin;
              delete curve 3 of trtagegraph;
              Rmax = R1;
             end;

       if R2 <= seuil
          then
            begin;
              type "Try to fit polynomial (third degree) ...";
            call $fitpolynomial("trtagegraph",1,3,,true,"anovapoly3",,,false);
              R3 = getcell("anovapoly3",1,7);
              if R3 > Rmax
                  then
                    begin;
                     delete curve 2 of trtagegraph;
                     Rmax = R3;
                     ajustement = "Apolynomial";
                    end;
                  else
                     delete curve 3 of trtagegraph;
              if R3 <= seuil
                 then
                    begin;
                       maxvalue = maximum(col 1 of trtage) - 1;
                       i = 0;
                       do j = 0 to 19;
                         set row 1 of col j of initvalues to (maxvalue + i);
                         i = i + 0.1;
                        end;
                     type "Try to fit [a*(1-exp(-b*X))] function ...";
                     call $fitfunction("A * ( 1 - exp(-B * X))","trtagegraph",
                                       1,,,5,0.001,"initvalues",fn,true,
                                      "tabresum",,,,false,false,,interfit,
                                      ,false);
                       R4 = getcell("tabresum",3,1);
                       if R4 > Rmax
                         then
                           begin;
```

```
                        delete curve 2 of trtagegraph;
                        Rmax = R4;
                        ajustement = "Afunction";
                      end;
                    else
                      delete curve 3 of trtagegraph;
                  delete tabresum;
                end;
          end;
     end;

set label of x axis of trtagegraph to "Age";
set libelle = encode(row 6 col 1 of inforecherche,"A(12)")
                    cat encode(row 12 col 1 of inforecherche,"A(8)");
set label of y axis of trtagegraph to libelle;
display trtagegraph bottomkey box height 0.9 width 0.7;

 */
splay row 1 to 10
    of col 0,1 of inforecherche where col 1 <> empty  at (61,0)
    nocolnumbers norownnumbers noheader notitle noheadingline ;
ll $measure(col 1 of trtage,"resumtab");
splay row 3,7,5 of col 1 of resumtab at (61,13)
    nocolnumbers norownnumbers noheader notitle noheadingline;

if yesanswer("Do you want to save the statistics ?")
   then
      begin;
        t = gettable("Name under which the table will be saved?",true);
        make table(t) from resumtab;
      end;

if yesanswer("Do you want an estimate of Y value for a given X ?")
   then
     begin;
       do while true;
       y = $evalcurve("trtagegraph",2);
       type nocr "y=", y;
       if yesanswer("Do you want to exit ?")
          then doexit;
     end;
 end;

if ajustement = "Alineaire"
  then
    begin;
      if yesanswer("Do you want add an confidence interval ?")
        then
          begin;
            call $_graph_ci("trtagegraph",1,,false);
            display trtagegraph bottomkey box height 0.9 width 0.7 at (0,0);
          end;
    end;

if yesanswer("Do you want to save this graph?")
   then
      begin;
        g = getgraph("Name under which the graph will be saved?",true);
        make graph(g) from trtagegraph;
      end;
```

```
  Deletion  */

delete trtagegraph;
if tableexists("anovapoly2")
   then
     delete anovapoly2;
if tableexists("anovapoly3")
   then
     delete anovapoly3;
if tableexists("anova")
   then
     delete table anova;
if tableexists("resumtab")
   then
     delete table resumtab;
if tableexists("tabresum")
   then
     delete table tabresum;
if tableexists("interfit")
   then
     delete interfit;


nd;
```

```
ocedure;

make graph trtpoidsgraph from col 0 of trtpoids vs col 1 of trtpoids;
type "Try to fit line...";
call $FITLINE("trtpoidsgraph", 1, equ ,true, "anova",,,false);
R1 = getcell("anova",1,7);
Seuil = getnumber("Give the threshold you want for fit : ");
ajustement = "Alineaire";
if R1 <= seuil
   then
     begin;
       type "Try to fit polynomial (second degree) ...";
       call $fitpolynomial("trtpoidsgraph",1,2,,true,"anovapoly2",,,false);
       R2 = getcell("anovapoly2",1,7);
       if R2 > R1
           then
             begin;
              delete curve 2 of trtpoidsgraph;
              Rmax = R2;
              ajustement = "Apolynomial";
             end;
          else
             begin;
              delete curve 3 of trtpoidsgraph;
              Rmax = R1;
             end;
       if R2 <= seuil
          then
            begin;
            type "Try to fit polynomial (third degree) ...";
            call $fitpolynomial("trtpoidsgraph",1,3,,true,"anovapoly3",,,false
;
              R3 = getcell("anovapoly3",1,7);
              if R3 > Rmax
                  then
                    begin;
                     delete curve 2 of trtpoidsgraph;
                     Rmax = R3;
                     ajustement = "Apolynomial";
                    end;
                 else
                     delete curve 3 of trtpoidsgraph;
              if R3 <= seuil
                 then
                    begin;
                       maxvalue = maximum(col 1 of trtpoids) - 1;
                       i = 0;
                       do j = 0 to 19;
                         set row 1 of col j of initvalues to (maxvalue + i);
                         i = i + 0.1;
                        end;
                     type "Try to fit  [a * (1-exp(-b*X))] function ...";
                     call $fitfunction("A * ( 1 - exp(-B * X))","trtpoidsgraph",
                                       1,,,5,0.001,"initvalues",fn,true,
                                      "tabresum",,,,false,
                                      false,,,,true);
                       R4 = getcell("tabresum",3,1);
                       if R4 > Rmax
                         then
                           begin;
                             delete curve 2 of trtpoidsgraph;
```

```
                                Rmax = R4;
                                ajustement = "Afunction";
                              end;
                            else
                              delete curve 3 of trtpoidsgraph;
                          delete tabresum;
                        end;
                end;
        end;
 set label of x axis of trtpoidsgraph to "Poids";
 set libelle = encode(row 6 col 1 of inforecherche,"A(12)")
                        cat encode(row 12 col 1 of inforecherche,"A(8)");
 set label of y axis of trtpoidsgraph to libelle;
 display trtpoidsgraph bottomkey box height 0.9 width 0.7;
  */
splay row 1 to 10
    of col 0,1 of inforecherche where col 1 <> empty  at (61,0)
    nocolnumbers norownnumbers noheader notitle noheadingline ;
ll $measure(col 1 of trtpoids,"resumtab");
splay row 3,7,5 of col 1 of resumtab at (61,13)
    nocolnumbers norownnumbers noheader notitle noheadingline;
 if yesanswer("Do you want to save this statistics ?")
    then
       begin;
         t = gettable("Name under which the table will be saved ?",true);
         make table(t) from resumtab;
       end;

if yesanswer("Do you want an estimate of Y value for a given X ?")
   then
     begin;
       do while true;
       y = $evalcurve("trtpoidsgraph",2);
       type nocr "y=", y;
       if yesanswer("Do you want to exit ?")
          then doexit;
     end;
 end;

f ajustement = "Alineaire"
 then
   if yesanswer("Do you want add an confidence interval ?")
     then
       begin;
          call $_graph_ci("trtpoidsgraph",1,,false);
          display trtpoidsgraph bottomkey box height 0.9 width 0.7 at (0,0);
       end;

 if yesanswer("Do you want to save this graph ?")
    then
       begin;
         g = getgraph("Name under which the graph will be saved ?",true);
         make graph(g) from trtpoidsgraph;
       end;
* deletion */
 delete trtpoidsgraph;
 if tableexists("anovapoly2")
    then
      delete anovapoly2 ;
 if tableexists("anovapoly3")
```

```
   then
      delete anovapoly3 ;
if tableexists("anova")
   then
      delete table anova ;
if tableexists("tabresum")
   then
      delete table tabresum ;
f tableexists("resumtab")
   then
      delete table resumtab ;
if tableexists("interfit")
   then
      delete interfit ;
d;
```

```
ROCEDURE(NOMTABLE,NOMGRAPHE,LIBELLE1,LIBELLE2);


*
   Réorganisation de la table des donnees pour distinguer en colonnes les
   différents [redacted] ECHANTILLONS.
/
  make table trt_init from table(NOMTABLE);
  X = "trt_init";
  ancien = getcell(X,1,0);
  i = 1;
  j = 2;
  insert col j of table(X) ;
  set row 0 col j of table(x) = ancien;
  for each row y in col 0 of table(X);
    nouveau = getcell(X,Y,0);
    if nouveau = ancien
      then
        begin;
          set row i col j of table(X) to col 1 row y;
          i = i + 1;
        end;
      else
        begin;
          i = 2;
          j = j + 1;
          insert col j of table(X);
          set row 0 col j of table(X) = nouveau;
          set row 1 col j of table(X) to col 1 row y;
          ancien = nouveau;
        end;
    end;

/*
Test de la normalite des  groupes
*/

/*
Cree une table contenant les resultats des tests de normalité pour chacun
des groupes (siglev: seuil)
*/

make table seuils from row 0 of col 0 of initvalues;
delete col 1 of seuils;

 SEUILINSERTION = 1;
 VARINSERTION = j + 1;
 DO k = 2 TO j;
  CALL $TESTNORMAL(COL k OF table(X),KWSTAT,SIGLEV,FALSE);
  NOMBRE = count of col k of table(x);
  if NOMBRE < 50
     then
       begin;
        VALEURTHEORIQUE = row NOMBRE col 2 of $@statcrit@w6;
        if VALEURTHEORIQUE > kwstat
           then
              GTEST = "SIG"; /* SIG = non normal */
           else
              GTEST = "NSIG"; /* NSIG = normal */
```

```
    end;
  else
    begin;
     VALEURDEPASSEMENT = 1 - $probchi(kwstat,4);
     if VALEURDEPASSEMENT < 0.05
        then
           GTEST = "SIG";
        else
           GTEST = "NSIG";
    end;

IF GTEST = "NSIG"
   THEN
    begin;
      SET GMOYENNE = MEAN OF COL k OF table(X) WHERE COL k <> EMPTY;
      SET GVARIANCE = VARIANCE OF COL k OF table(X) WHERE COL k <> EMPTY;
      SET Z = 1;
      INSERT COL VARINSERTION OF table(X);
      FOR EACH ROW Y IN COL k OF table(X);
        SET E1 = SQRT(GVARIANCE) * SQRT(2*3.14159);
        VALEURDEX = GETCELL(X,Y,k);
        IF VALEURDEX <> EMPTY
          THEN
            BEGIN;
              SET E2 = -((VALEURDEX - GMOYENNE)**2);
              SET E3 = 2 * GVARIANCE;
              SET FDEX = (1/E1) * (EXP(E2/E3));
              SET ROW Z COL VARINSERTION OF table(X) TO FDEX;
            END;
        SET Z = Z + 1;
      END;
      VARINSERTION = VARINSERTION + 1;
    END;
  insert col SEUILINSERTION of seuils;
  set row 0 col seuilinsertion of seuils to row 0 col k of table(X);
  set row 1 col seuilinsertion of seuils to GTEST ;
  SEUILINSERTION = SEUILINSERTION + 1;

END;

Dessin du graphe */

NONNORMAL = "NON";
gaussienne = j + 1;
DO i = 1 to j - 1;
   if row 1 col i of seuils = "NSIG"
      then
        begin;
             TEMP = i + 1;
             MAKE GRAPH (NOMGRAPHE) FROM COL TEMP OF table(X)
                                    VS COL gaussienne of table(X);
             gaussienne = gaussienne + 1;
             DO v = i + 1  to j - 1;
                if row 1 col v of seuils = "NSIG"
                   then
                     BEGIN;
                         ADD CURVE TO graph(NOMGRAPHE) FROM COL (v + 1) OF
                              table(X) VS COL gaussienne of table(X);
                         gaussienne = gaussienne + 1;
                     END;
```

```
                    else
                      NONNORMAL = "OUI";
                END; /* END DO v */
               graphexiste = "true";
               doexit;
          END;
        else
            NONNORMAL = "OUI";

  END; /* END DO i */

 Libelle des courbes */

 if graphexiste = "true"
    then
      begin;
        set libelle = encode(row 6 col 1 of inforecherche,"A(12)")
                 cat encode(row 12 col 1 of inforecherche,"A(8)");
        set label of x axis of graph(NOMGRAPHE) to libelle;
        NUMCOURBES = 1;
        DO c = 1 to j - 1;
           if row 1 col c of seuils = "NSIG"
             then
               begin;
                 set label of curve NUMCOURBES of graph(NOMGRAPHE)
                                  to row 0 col c of seuils;
                 NUMCOURBES = NUMCOURBES + 1;
               end;

         end;

* Affichage du graphe et des echantillons non normaux(s'il y en a) */

        DISPLAY graph(NOMGRAPHE) bottomkey height 0.9 width 0.7;
        DO l = 1 to j - 1;
          if row 1 col l of seuils = "SIG"
           then
        type nocr "The sample ",row 0 col l of seuils
                  ," hasn't a normal distribution !";
         end;
     end;
*

alcul des statistiques et affichage des informations de la recherche
a table calculstat contient les valeurs calculées des statistiques demandées.
lle sert à afficher tous les résultats en une fois. Elle doit TOUJOURS
igurer dans la directory RS1 .

/

ake table calculstat_prov from col 0 of calculstat;
elete col 1 of calculstat_prov;
nsert row 4 of calculstat_prov;
nsert row 5 of calculstat_prov;
et row 4 col 0 of calculstat_prov = "Low limit";
et row 5 col 0 of calculstat_prov = "Up limit";
et row 1 col 0 of calculstat_prov = "Mean";
et row 2 col 0 of calculstat_prov = "S.D.";
et row 3 col 0 of calculstat_prov = "Median";
 = 1;
```

```
 m = 2 to J;
set gmediane = median of col m of table(X) where col m <> empty;
set gmoyenne = mean of col m of table(X) where col m <> empty;
set gvariance = variance of col m of table(X) where col m <> empty;
insert col n of calculstat_prov;
set row 0 col n of calculstat_prov = row 0 col m of table(X);
set row 1 col n of calculstat_prov = gmoyenne;
set row 2 col n of calculstat_prov = sqrt(gvariance);
set row 3 col n of calculstat_prov = gmediane;

 Ajout des valeurs limites */

SAMPLENAME = row 0 col m of table(x);
call VALEURDEREF(row 1 col (m - 1) of seuils,gmoyenne,sqrt(gvariance),
                 x ,UPLIMIT,LOWLIMIT,SEUIL,m,SAMPLENAME);
LOW = encode(SEUIL,"A(4)") cat encode(LOWLIMIT,"A(5)");
UP = encode(SEUIL,"A(4)") cat encode(UPLIMIT,"A(5)");
set row 4 col n of calculstat_prov = LOW;
set row 5 col n of calculstat_prov = UP;
 FIN valeurs limites */

n = n + 1;
ND;

splay row 1 to 10 of col 0,1 of inforecherche where col 1 <> empty
          at (61,0) nocolnumbers norownumbers noheader notitle noheadingline;
splay table calculstat_prov at (54,12) nocolnumbers norownumbers noheader
                notitle noheadingline;

 graphexiste = "true"
then
  if yesanswer("Do you want to save this graph ?")
    then
      begin;
        g = getgraph("Name under which the graph will be saved ?",N);
        make graph(g) from graph(NOMGRAPHE);
      end;

f yesanswer("Do you want to watch the whole statistic table ?")
  then
    begin;
      call erase;
      display table calculstat_prov nocolnumbers norownumbers;
    end;
f yesanswer("Do you want to save this table ?")
  then
   begin;
    t = gettable("Name under which the table will be saved ?",N);
    make table(t) from calculstat_prov;
   end;

emande d'affichage d'histogrammes pour les echantillons non normaux
/

f NONNORMAL = "OUI"
  then
f yesanswer("Do you want to watch the histograms for non normal samples ?")
hen
 begin;
```

```
  set libel1 = encode(row 6 col 1 of inforecherche,"A(12)")
               cat encode(row 12 col 1 of inforecherche,"A(8)");
  call erase(true);
  DO l = 1 to j - 1;
    if row 1 col l of seuils = "SIG"
      then
        begin;
          type nocr "Histogram for ",row 0 col (l+1) of table(x)," : ";
          set borninf = minimum of col (l+1) of table(x);
          set bornsup = maximum of col (l+1) of table(x);
          continuer = "true";
          set libelle = encode(libel1,"A(20)") cat encode(row 0 col (l+1) of
                        table(x),"A(15)");
          DO WHILE continuer = "true";
            nbinterval = getnumber("Number of intervals ?",false);
            call erase;
            call $histogram(col (l + 1) of table(x),"histographe",false,
                            borninf,bornsup,nbinterval);
            set label of x axis of histographe to libelle;
            display graph histographe bottomkey height 0.9 width 0.7;
            display row 1 to 10 of col 0,1 of inforecherche where col 1 <>
                          EMPTY at (61,0) nocolnumbers norownumbers noheader
                          notitle noheadingline;
            if yesanswer("Do you want change the intervals number ?")
              then
                begin;
                  continuer = "true";
                  delete graph histographe;
                end;
              else
                 continuer = "false";
           end;
          if yesanswer("Do you want to save this histogram ?")
           then
            begin;
             g = getgraph("Name under which the histogram will be saved ?",N);
             make graph(g) from histographe;
            end;
          delete graph histographe;
       end;
    end;
end;


*
Test de l'égalité des moyennes si l'utilisateur le désire (95 %)
/

 if yesanswer(LIBELLE1)
     then
       begin;
         CALL $ANOVAONEWAY(COL 1 OF table(X),COL 0 OF table(X),"tabanova"
                           ,false,,"resume");
         if row 1 col 5 of tabanova < 0.05
           then
            BEGIN;
            type LIBELLE2 ;
            if yesanswer("Do you want to determine where the difference lie?")
               then
                 begin;
```

```
                CALL $SIMULTANEOUS("resume","DUNCAN",,,,"compmult",false);
                display compmult;
                if yesanswer("Do you want to save this table?")
                   then
                     begin;
                t = gettable("Name under which the table will be saved ?",N);
                  make table(t) from compmult;
                     end;
              end;
          END;
       end;


Effacage des tables


delete seuils;
delete trt_init;
delete calculstat_prov;
delete graph(NOMGRAPHE);
if tableexists("tabanova")
   then
     delete tabanova;
if tableexists("resume")
   then
     delete resume;
if tableexists("compmult")
   then
    delete compmult;
ID;
```

```
ocedure;

make graph evopoidsgraph from col 0 of evopoids vs col 1 of evopoids;
* Initialisation de la table contenant les valeurs des parametres A et B
  à estimer
/
     maxvalue = maximum(col 1 of evopoids) - 1;
     i = 0;
     do j = 0 to 19;
          set row 1 of col j of initvalues to (maxvalue + i);
          i = i + 0.1;
     end;

 Intervalle de confiance */


if yesanswer("Do you want to see the confidence intervals ?")
   then
     begin;
       type "Try to fit [a*(1-exp(-B*X))] function ...";
       call $fitfunction("A * ( 1 - exp(-B * X))","evopoidsgraph",
                         1,,,5,0.001,"initvalues",fn,true,
                         "tabresum",,,,false,false,,,
                          ,true);
     end;
   else
     begin;
       type "Try to fit [a*(1-exp(-B*X))] function ...";
       call $fitfunction("A * ( 1 - exp(-B * X))","evopoidsgraph",
                         1,,,5,0.001,"initvalues",fn,true,
                         "tabresum",,,,false,false,,,
                          ,false);
     end;
set label of x axis of evopoidsgraph to "Day of study";
set label of y axis of evopoidsgraph to "Weight    gr ";
set label of curve 1 of evopoidsgraph to DATA;
display evopoidsgraph bottomkey box height 0.9 width 0.7;

* */
isplay row 1
     of col 0,1 of RECHERCHETOXICO where col 1 <> empty  at (61,0)
     nocolnumbers norownnumbers noheader notitle noheadingline ;
all $measure(col 1 of evopoids,"resumtab");
isplay row 3,7,5 of col 1 of resumtab at (61,13)
     nocolnumbers norownnumbers noheader notitle noheadingline;

  if yesanswer("Do you want to save this statistics ?")
     then
        begin;
          t = gettable("Name under which the table will be saved ?",True);
          make table(t) from resumtab;
        end;

* Demande des points pour un animal */

 if yesanswer
"Do you want to compare the evolution
 of an animal with the reference sample?")
    then
      BEGIN;
```

```
   if yesanswer("Enter the data from a table ?")
      then
        begin;
          t = gettable("Name of the table to use: ");
          DO while not tableexists(t);
             type "ERROR this table do not exist !!!";
             t = gettable("Name of the table to use: ");
          end;
          add curve to evopoidsgraph from col 0 of table(t)
                    VS col 1 of table(t);
        end;
      else
        begin;
          make table ANIMAL from col 0 of calculstat;
          delete row 1,2,3 of ANIMAL;
          i = 1;
          CONTINUER = true;
          DO while CONTINUER ;

            set JOURETUDE = GETNUMBER("Day of study : ",TRUE);
            set VALEUR = GETNUMBER("Weight : ",TRUE);
            if (JOURETUDE = empty) or (VALEUR = empty)
               then
                  CONTINUER = false;
               else
                 begin;
                   insert row i of ANIMAL;
                   set row i col 0 of ANIMAL to JOURETUDE;
                   set row i col 1 of ANIMAL to VALEUR;
                   i = i + 1;
                 end;
          END;
          add curve to evopoidsgraph from col 0 of ANIMAL
                    VS col 1 of ANIMAL;
          if yesanswer("Do you want to save this table?")
             then
                begin;
                  t = gettable("Name of the table : ",True);
                  make table(t) from ANIMAL;
                end;
        end;
    set label of curve 3 of evopoidsgraph to "Compared animal curve";
    display evopoidsgraph bottomkey height 0.9 width 0.65 at (0,0);
  END;


* Estimation de Y à partir de X */

 if yesanswer("Do you want an estimate of Y value for a given X ?")
    then
      begin;
        do while true;
           y = $evalcurve("evopoidsgraph",2);
           type nocr "y=", y;
           if yesanswer("Do you want to exit ?")
              then doexit;
        end;
      end;
```

```
 if yesanswer("Do you want to save this graph ?")
    then
       begin;
         g = getgraph("Name under which the graph will be saved ?",True);
         make graph(g) from evopoidsgraph;
       end;

 deletion */

delete evopoidsgraph;
if tableexists("tabresum")
   then
      delete table tabresum ;
if tableexists("resumtab")
   then
      delete table resumtab ;
if tableexists("ANIMAL")
   then
      delete ANIMAL ;

id;
```

```
ocedure;


 Reorganisation de la table des données pour distinguer en colonnes les
 différents sexes

make table evopetsexe_init from evopetsexe;
x = "evopetsexe_init";
ancien = getcell(x,1,2);
i = 1;
j = 3;
jbis = j + 1;
insert col j of table(x);
insert col jbis of table(x);
set row 0 col j of table(x) = ancien;
FOR EACH ROW y IN col 2 of table(x);
    nouveau = getcell(x,y,2);
    if nouveau = ancien
       then
          begin;
            set row i col j of table(x) to col 0 row y;
            set row i col jbis of table(x) to col 1 row y;
            i = i + 1;
          end;
       else
          begin;
            i = 2;
            j = j + 2;
            jbis = j + 1;
            insert col j of table(x);
            insert col jbis of table(x);
            set row 0 col j of table(x) = nouveau;
            set row 1 col j of table(x) to col 0 row y;
            set row 1 col jbis of table(x) to col 1 row y;
            ancien = nouveau;
          end;
  end;


make graph evopetsexegraph from col 3 of evopetsexe_init
                          vs col 4 of evopetsexe_init;
add curve to evopetsexegraph from col 5 of evopetsexe_init
                          vs col 6 of evopetsexe_init;


/* Initialisation de la table contenant les valeurs des parametres A et B
   à estimer pour les femelles
*/
      maxvalue = maximum(col 4 of evopetsexe_init) - 1;
      i = 0;
      do j = 0 to 19;
           set row 1 of col j of initvalues to (maxvalue + i);
           i = i + 0.1;
      end;

* intervalle de confiance */

 if yesanswer("Do you want to see the confidence intervals ?")
    then
     begin;
```

```
    reponse = gettext("For males(M), females(F) or twice(T) ?");
    do while (reponse <> "M") and (reponse <> "F") and (reponse <> "T");
       type "ERROR You must enter M,F or T";
       reponse = gettext("For males(M), females(F) or twice(T) ?");
     end;
   end;
  else
    set reponse = "R";
pe "Try to fit [a*(1-exp(-b*X))] function for females...";
 (reponse = "F") or (reponse = "T")
 then
     call $fitfunction("A * ( 1 - exp(-B * X))","evopetsexegraph",
                       1,,,5,0.001,"initvalues",fn,true,
                       "tabresum",,,,false,false,,,
                        ,true);
  else
     call $fitfunction("A * ( 1 - exp(-B * X))","evopetsexegraph",
                       1,,,5,0.001,"initvalues",fn,true,
                       "tabresum",,,,false,false,,,
                        ,false);
set label of x axis of evopetsexegraph to "Day of study";
set label of y axis of evopetsexegraph to "Weight    gr ";
set label of curve 1 of evopetsexegraph to "DATA (female)";

 Initialisation de la table contenant les valeurs des parametres A et B
  à estimer pour les males
/
    maxvalue = maximum(col 6 of evopetsexe_init) - 1;
    i = 0;
    do j = 0 to 19;
        set row 1 of col j of initvalues to (maxvalue + i);
        i = i + 0.1;
    end;

type "Try to fit [a*(1-exp(-b*X))] function for males...";
if (reponse = "M") or (reponse = "T")
 then
     call $fitfunction("A * ( 1 - exp(-B * X))","evopetsexegraph",
                       2,,,5,0.001,"initvalues",fn,true,
                       "tabresum2",,,,false,false,,,
                        ,true);
  else
    call $fitfunction("A * ( 1 - exp(-B * X))","evopetsexegraph",
                       2,,,5,0.001,"initvalues",fn,true,
                       "tabresum2",,,,false,false,,,
                        ,false);

set label of curve 2 of evopetsexegraph to "DATA (male)";

display evopetsexegraph bottomkey box height 0.9 width 0.7;


 */
splay row 1
    of col 0,1 of RECHERCHETOXICO where col 1 <> empty  at (61,0)
    nocolnumbers norownumbers noheader notitle noheadingline ;
t row 1 col 0 of calculstat = "Mean";
t row 2 col 0 of calculstat = "S.D.";
t row 3 col 0 of calculstat = "Median";
t row 0 col 1 of calculstat = "F";
```

```
t row 1 col 1 of calculstat = mean of col 4 of evopetsexe_init;
t row 2 col 1 of calculstat = sqrt(variance of col 4 of evopetsexe_init);
t row 3 col 1 of calculstat = median of col 4 of evopetsexe_init;
t row 0 col 2 of calculstat = "M";
t row 1 col 2 of calculstat = mean of col 6 of evopetsexe_init;
t row 2 col 2 of calculstat = sqrt(variance of col 6 of evopetsexe_init);
t row 3 col 2 of calculstat = median of col 6 of evopetsexe_init;

splay table calculstat at (58,13)
    nocolnumbers norownumbers noheader notitle noheadingline;

 if yesanswer("Do you want to save this statistics ?")
    then
       begin;
         t = gettable("Name under which the table will be saved ?",True);
         make table(t) from calculstat;
       end;

 Demande des points pour un animal */

if yesanswer
Do you want to compare the evolution
of an animal with the reference sample?")
   then
     BEGIN;
      if yesanswer("Enter the data from a table ?")
         then
           begin;
             t = gettable("Name of the table to use: ");
             DO while not tableexists(t);
                type "ERROR this table do not exist !!!";
                t = gettable("Name of the table to use: ");
             end;
             add curve to evopetsexegraph from col 0 of table(t)
                       VS col 1 of table(t);
           end;
         else
           begin;
             make table ANIMAL from col 0 of calculstat;
             delete row 1,2,3 of ANIMAL;
             i = 1;
             CONTINUER = true;
             DO while CONTINUER ;

               set JOURETUDE = GETNUMBER("Day of study : ",TRUE);
               set VALEUR = GETNUMBER("Weight : ",TRUE);
               if (JOURETUDE = empty) or (VALEUR = empty)
                  then
                     CONTINUER = false;
                  else
                    begin;
                      insert row i of ANIMAL;
                      set row i col 0 of ANIMAL to JOURETUDE;
                      set row i col 1 of ANIMAL to VALEUR;
                      i = i + 1;
                    end;
             END;
             add curve to evopetsexegraph from col 0 of ANIMAL
                       VS col 1 of ANIMAL;
             if yesanswer("Do you want to save this table?")
```

```
                then
                   begin;
                     t = gettable("Name of the table : ",True);
                     make table(t) from ANIMAL;
                   end;
          end;
       set label of curve 5 of evopetsexegraph to "Compared animal curve";
       display evopetsexegraph bottomkey height 0.9 width 0.7 at (0,0);
     END;

 Estimation de Y à partir de X */

if yesanswer("Do you want an estimate of Y value for a given X ?")
   then
     begin;
       if yesanswer("For Male ?")
          then
            begin;
              if (reponse = "F") or (reponse = "T")
                 then
                   numcourbe = 6;
                 else
                   numcourbe = 4;
              do while true;
                y = $evalcurve("evopetsexegraph",numcourbe);
                type nocr "y=", y;
                if yesanswer("Do you want to exit ?")
                   then doexit;
              end;
            end;
       if yesanswer("For Female ?")
          then
            begin;
              do while true;
                y = $evalcurve("evopetsexegraph",3);
                type nocr "y=", y;
                if yesanswer("Do you want to exit ?")
                   then doexit;
              end;
            end;
     end;


 if yesanswer("Do you want to save this graph ?")
    then
       begin;
         g = getgraph("Name under which the graph will be saved ?",True);
         make graph(g) from evopetsexegraph;
       end;

 deletion */

delete evopetsexegraph;
delete evopetsexe_init;
if tableexists("tabresum")
   then
      delete table tabresum ;
if tableexists("interfit")
   then
      delete interfit ;
```

```
if tableexists("tabresum2")
   then
      delete table tabresum2 ;
if tableexists("interfit2")
   then
      delete interfit2 ;
if tableexists("ANIMAL")
   then
      delete ANIMAL ;

d;
```

```
cedure;

ake graph evopagegraph from col 0 of evopage vs col 1 of evopage;
 Initialisation de la table contenant les valeurs des parametres A et B
 à estimer

     maxvalue = maximum(col 1 of evopage) - 1;
     i = 0;
     do j = 0 to 19;
          set row 1 of col j of initvalues to (maxvalue + i);
          i = i + 0.1;
     end;

Intervalle de confiance */


f yesanswer("Do you want to see the confidence intervals ?")
   then
     begin;
       intervalle = "y";
       type "Try to fit [a*(1-exp(-B*X))] function ...";
       call $fitfunction("A * ( 1 - exp(-B * X))","evopagegraph",
                         1,,,5,0.001,"initvalues",fn,true,
                         "tabresum",,,,false,false,,,
                          ,true);
     end;
   else
     begin;
       intervalle = "n";
       type "Try to fit [a*(1-exp(-B*X))] function ...";
       call $fitfunction("A * ( 1 - exp(-B * X))","evopagegraph",
                         1,,,5,0.001,"initvalues",fn,true,
                         "tabresum",,,,false,false,,,
                          ,false);
     end;
set label of x axis of evopagegraph to "Age (days)";
set label of y axis of evopagegraph to "Weight    gr ";
set label of curve 1 of evopagegraph to DATA;
display evopagegraph bottomkey box height 0.9 width 0.7;

 */
splay row 1
    of col 0,1 of RECHERCHETOXICO where col 1 <> empty  at (61,0)
    nocolnumbers norownnumbers noheader notitle noheadingline ;
ll $measure(col 1 of evopage,"resumtab");
splay row 3,7,5 of col 1 of resumtab at (61,13)
    nocolnumbers norownnumbers noheader notitle noheadingline;

 if yesanswer("Do you want to save this statistics ?")
    then
       begin;
         t = gettable("Name under which the table will be saved ?",True);
         make table(t) from resumtab;
       end;

 Demande des points pour un animal */

if yesanswer
Do you want to compare the evolution
of an animal with the reference sample?")
```

```
   then
     BEGIN;
      if yesanswer("Enter the data from a table ?")
         then
           begin;
             t = gettable("Name of the table to use: ");
             DO while not tableexists(t);
                type "ERROR this table do not exist !!!";
                t = gettable("Name of the table to use: ");
             end;
             add curve to evopagegraph from col 0 of table(t)
                       VS col 1 of table(t);
           end;
         else
           begin;
             make table ANIMAL from col 0 of calculstat;
             delete row 1,2,3 of ANIMAL;
             i = 1;
             CONTINUER = true;
             DO while CONTINUER ;

               set AGE = GETNUMBER("Age (days) : ",TRUE);
               set VALEUR = GETNUMBER("Weight : ",TRUE);
               if (AGE = empty) or (VALEUR = empty)
                  then
                     CONTINUER = false;
                  else
                    begin;
                      insert row i of ANIMAL;
                      set row i col 0 of ANIMAL to AGE;
                      set row i col 1 of ANIMAL to VALEUR;
                      i = i + 1;
                    end;
             END;
             add curve to evopagegraph from col 0 of ANIMAL
                       VS col 1 of ANIMAL;
             if yesanswer("Do you want to save this table?")
                then
                   begin;
                     t = gettable("Name of the table : ",True);
                     make table(t) from ANIMAL;
                   end;
           end;
       if intervalle = "y"
         then
           set label of curve 5 of evopagegraph to "Compared animal curve";
         else
           set label of curve 3 of evopagegraph to "Compared animal curve";
       display evopagegraph bottomkey height 0.9 width 0.65 at (0,0);
     END;


/* Estimation de Y à partir de X */

  if yesanswer("Do you want an estimate of Y value for a given X ?")
     then
       begin;
         do while true;
            y = $evalcurve("evopagegraph",2);
```

```
          type nocr "y=", y;
          if yesanswer("Do you want to exit ?")
             then doexit;
       end;
     end;


 if yesanswer("Do you want to save this graph ?")
    then
       begin;
         g = getgraph("Name under which the graph will be saved ?",True);
         make graph(g) from evopagegraph;
       end;

 deletion */

delete evopagegraph;
if tableexists("tabresum")
   then
      delete table tabresum ;
if tableexists("resumtab")
   then
      delete table resumtab ;
if tableexists("ANIMAL")
   then
      delete ANIMAL ;

nd;
```

```
cedure;


 Reorganisation de la table des données pour distinguer en colonnes les
 différents races

iake table evoprace_init from evoprace;
: = "evoprace_init";
incien = getcell(x,1,2);
. = 1;
j = 3;
jbis = j + 1;
.nsert col j of table(x);
.nsert col jbis of table(x);
set row 0 col j of table(x) = ancien;
FOR EACH ROW y IN col 2 of table(x);
    nouveau = getcell(x,y,2);
    if nouveau = ancien
       then
          begin;
            set row i col j of table(x) to col 0 row y;
            set row i col jbis of table(x) to col 1 row y;
            i = i + 1;
          end;
       else
          begin;
            i = 2;
            j = j + 2;
            jbis = j + 1;
            insert col j of table(x);
            insert col jbis of table(x);
            set row 0 col j of table(x) = nouveau;
            set row 1 col j of table(x) to col 0 row y;
            set row 1 col jbis of table(x) to col 1 row y;
            ancien = nouveau;
          end;
  end;


 dessin des courbes */

ke graph evopracegraph from col 3 of table(x)
                          vs col 4 of table(x);
belle = encode("Data from race ","A(15)") cat
          encode(row 0 col 3 of table(x),"A(15)");
t label of curve 1 of evopracegraph to libelle ;
= 5;
urbe = 2;
 WHILE i <= j;
 ibis = i + 1;
 libelle = encode("Data from race ","A(15)") cat
           encode(row 0 col i of table(x),"A(15)");
 ADD curve to evopracegraph from col i of table(x) vs col ibis of table(x);
 set label of curve courbe of evopracegraph to libelle ;
 courbe = courbe + 1;
 i = i + 2;
end;
```

```
 intervalle de confiance */

if yesanswer("Do you want to see the confidence intervals ?")
   then
       INTERVALLE = "OUI";
   else
       INTERVALLE = "NON";

 Initialisation de la table contenant les valeurs des parametres A et B
  à estimer
/

= 3;
rvenumber = 1;
 WHILE i <= j;
ibis = i + 1;
maxvalue = maximum(col ibis of table(x)) - 1;
K = 0;
do l = 0 to 19;
   set row 1 of col 1 of initvalues to (maxvalue + k);
   k = k + 0.1;
 end;
type nocr "Try to fit [a*(1-exp(-b*X))] function for race ",row 0 col i
                                  of table(x);
type nocr " ...";
if intervalle = "OUI"
  then
     call $fitfunction("A * ( 1 - exp(-B * X))","evopracegraph",
                       curvenumber,,,5,0.001,"initvalues",,true,
                       ,,,,false,false,,,
                        ,true);
   else
      call $fitfunction("A * ( 1 - exp(-B * X))","evopracegraph",
                       curvenumber,,,5,0.001,"initvalues",,true,
                       ,,,,false,false,,,
                        ,false);
  i = i + 2;
  curvenumber = curvenumber + 1;
id;

set label of x axis of evopracegraph to "Age (days)";
set label of y axis of evopracegraph to "Weight    gr ";


* libelle des courbes */

BRACE = (jbis - 2)/2;
 = 3;
) WHILE i <= j;
ibis = i + 1;
libelle = encode("Fitted function for ","A(20)") cat
          encode(row 0 col i of table(x),"A(15)");
if intervalle = "OUI"
  then
    set label of curve ((((3*ibis)/2)-5)+NBRACE) of evopracegraph to
        libelle ;
  else
    set label of curve (((ibis - 2)/2)+NBRACE) of evopracegraph to
        libelle ;
i = i + 2;
```

```
;

isplay evopracegraph bottomkey box height 0.9 width 0.65;


*/
play row 1
    of col 0,1 of RECHERCHETOXICO where col 1 <> empty  at (61,0)
    nocolnumbers norownumbers noheader notitle noheadingline ;

e table calculstat_prov from col 0 of calculstat;
ete col 1 of calculstat_prov;
 row 1 col 0 of calculstat_prov = "Mean";
 row 2 col 0 of calculstat_prov = "S.D.";
 row 3 col 0 of calculstat_prov = "Median";
 1;
 3;
WHILE m <= j;
insert col n of calculstat_prov;
mbis = m + 1;
set row 0 col n of calculstat_prov to row 0 col m of table(x);
set row 1 col n of calculstat_prov = mean of col mbis of table(x);
set row 2 col n of calculstat_prov = sqrt(variance of col mbis of table(x));
set row 3 col n of calculstat_prov = median of col mbis of table(x);
m = m + 2;
n = n + 1;
d;

splay table calculstat_prov at (58,13)
    nocolnumbers norownumbers noheader notitle noheadingline;

yesanswer("Do you want to watch the whole statistic table ?")
then
begin;
  call erase;
  display table calculstat_prov nocolnumbers norownumbers;
end;

yesanswer("Do you want to save this table ?")
then
  begin;
    t = gettable("Name under which the table will be saved ?",True);
    make table(t) from calculstat_prov;
  end;

Demande des points pour un animal */

if yesanswer
Do you want to compare the evolution
of an animal with the reference sample?")
   then
     BEGIN;
      if yesanswer("Enter the data from a table ?")
         then
           begin;
             t = gettable("Name of the table to use: ");
             DO while not tableexists(t);
                type "ERROR this table do not exist !!!";
                t = gettable("Name of the table to use: ");
             end;
```

```
            add curve to evopracegraph from col 0 of table(t)
                      VS col 1 of table(t);
          end;
        else
          begin;
            make table ANIMAL from col 0 of calculstat;
            delete row 1,2,3 of ANIMAL;
            i = 1;
            CONTINUER = true;
            DO while CONTINUER ;

              set AGE = GETNUMBER("Age (days) : ",TRUE);
              set VALEUR = GETNUMBER("Weight : ",TRUE);
              if (AGE = empty) or (VALEUR = empty)
                 then
                    CONTINUER = false;
                 else
                   begin;
                     insert row i of ANIMAL;
                     set row i col 0 of ANIMAL to AGE;
                     set row i col 1 of ANIMAL to VALEUR;
                     i = i + 1;
                   end;
            END;
            add curve to evopracegraph from col 0 of ANIMAL
                       VS col 1 of ANIMAL;
            if yesanswer("Do you want to save this table?")
               then
                  begin;
                    t = gettable("Name of the table : ",True);
                    make table(t) from ANIMAL;
                  end;
          end;
      if intervalle = "OUI"
        then
         set label of curve (((jbis - 2)*2)+1) of evopracegraph
                     to "Compared animal curve";
        else
         set label of curve ((jbis - 2) + 1) of
                            evopracegraph to "Compared animal curve";
      display evopracegraph bottomkey height 0.9 width 0.65 at (0,0);
    END;

* Estimation de Y à partir de X */

if yesanswer("Do you want an estimate of Y value for a given X ?")
   then
     begin;
       i = 3;
       DO WHILE i <= j;
         ibis = i + 1;
         QUEST = encode("For ","A(4)") cat
                 encode (row 0 col i of table(x),"A(15)");
         QUESTION = encode(QUEST,"A(19)") cat encode(" ?","A(2)");
         if yesanswer(QUESTION)
           then
             begin;
               if intervalle = "OUI"
                 then
                   numcourbe = ((((3*ibis)/2)-5)+NBRACE);
```

```
                else
                  numcourbe = (((ibis - 2)/2)+NBRACE);
              do while true;
                y = $evalcurve("evopracegraph",numcourbe);
                type nocr "y=", y;
                if not yesanswer("For an another point for this race ?")
                   then doexit;
              end;
            end;
          if not yesanswer("For an another race ?")
            then doexit;
          i = i + 2;
        end;
    end;

 if yesanswer("Do you want to save this graph ?")
    then
       begin;
         g = getgraph("Name under which the graph will be saved ?",True);
         make graph(g) from evopracegraph;
       end;

 deletion */

delete evopracegraph;
delete calculstat_prov;
delete evoprace_init;
if tableexists("ANIMAL")
   then
      delete ANIMAL ;
d;
```

```
ocedure;


 Reorganisation de la table des données pour distinguer en colonnes les
 différents sexes

make table evopsexe_init from evopsexe;
x = "evopsexe_init";
ancien = getcell(x,1,2);
i = 1;
j = 3;
jbis = j + 1;
insert col j of table(x);
insert col jbis of table(x);
set row 0 col j of table(x) = ancien;
FOR EACH ROW y IN col 2 of table(x);
    nouveau = getcell(x,y,2);
    if nouveau = ancien
       then
          begin;
            set row i col j of table(x) to col 0 row y;
            set row i col jbis of table(x) to col 1 row y;
            i = i + 1;
          end;
       else
          begin;
            i = 2;
            j = j + 2;
            jbis = j + 1;
            insert col j of table(x);
            insert col jbis of table(x);
            set row 0 col j of table(x) = nouveau;
            set row 1 col j of table(x) to col 0 row y;
            set row 1 col jbis of table(x) to col 1 row y;
            ancien = nouveau;
          end;
  end;


nake graph evopsexegraph from col 3 of evopsexe_init
                          vs col 4 of evopsexe_init;
add curve to evopsexegraph from col 5 of evopsexe_init
                          vs col 6 of evopsexe_init;


/* Initialisation de la table contenant les valeurs des parametres A et B
   à estimer pour les femelles
*/
      maxvalue = maximum(col 4 of evopsexe_init) - 1;
      i = 0;
      do j = 0 to 19;
           set row 1 of col j of initvalues to (maxvalue + i);
           i = i + 0.1;
      end;

* intervalle de confiance */

 if yesanswer("Do you want to see the confidence intervals ?")
    then
     begin;
```

```
    reponse = gettext("For males(M), females(F) or twice(T) ?");
    do while (reponse <> "M") and (reponse <> "F") and (reponse <> "T");
       type "ERROR You must enter M,F or T";
       reponse = gettext("For males(M), females(F) or twice(T) ?");
     end;
   end;
  else
    set reponse = "R";
pe "Try to fit [a*(1-exp(-b*X))] function for females...";
 (reponse = "F") or (reponse = "T")
 then
     call $fitfunction("A * ( 1 - exp(-B * X))","evopsexegraph",
                       1,,,5,0.001,"initvalues",fn,true,
                       "tabresum",,,,false,false,,,
                        ,true);
  else
     call $fitfunction("A * ( 1 - exp(-B * X))","evopsexegraph",
                       1,,,5,0.001,"initvalues",fn,true,
                       "tabresum",,,,false,false,,,
                        ,false);
et label of x axis of evopsexegraph to "Age (days)";
et label of y axis of evopsexegraph to "Weight    gr ";
et label of curve 1 of evopsexegraph to "DATA (female)";

Initialisation de la table contenant les valeurs des parametres A et B
 à estimer pour les males

    maxvalue = maximum(col 6 of evopsexe_init) - 1;
    i = 0;
    do j = 0 to 19;
         set row 1 of col j of initvalues to (maxvalue + i);
         i = i + 0.1;
    end;

ype "Try to fit [a*(1-exp(-b*X))] function for males...";
f (reponse = "M") or (reponse = "T")
 then
     call $fitfunction("A * ( 1 - exp(-B * X))","evopsexegraph",
                       2,,,5,0.001,"initvalues",fn,true,
                       "tabresum2",,,,false,false,,,
                        ,true);
  else
    call $fitfunction("A * ( 1 - exp(-B * X))","evopsexegraph",
                       2,,,5,0.001,"initvalues",fn,true,
                       "tabresum2",,,,false,false,,,
                        ,false);

set label of curve 2 of evopsexegraph to "DATA (male)";

display evopsexegraph bottomkey box height 0.9 width 0.7;


*/
splay row 1
   of col 0,1 of RECHERCHETOXICO where col 1 <> empty  at (61,0)
   nocolnumbers norownumbers noheader notitle noheadingline ;
 row 1 col 0 of calculstat = "Mean";
 row 2 col 0 of calculstat = "S.D.";
 row 3 col 0 of calculstat = "Median";
 row 0 col 1 of calculstat = "F";
```

```
t row 1 col 1 of calculstat = mean of col 4 of evopsexe_init;
t row 2 col 1 of calculstat = sqrt(variance of col 4 of evopsexe_init);
t row 3 col 1 of calculstat = median of col 4 of evopsexe_init;
t row 0 col 2 of calculstat = "M";
t row 1 col 2 of calculstat = mean of col 6 of evopsexe_init;
t row 2 col 2 of calculstat = sqrt(variance of col 6 of evopsexe_init);
t row 3 col 2 of calculstat = median of col 6 of evopsexe_init;

splay table calculstat at (58,13)
    nocolnumbers norownumbers noheader notitle noheadingline;

 if yesanswer("Do you want to save this statistics ?")
    then
       begin;
         t = gettable("Name under which the table will be saved ?",True);
         make table(t) from calculstat;
       end;

 Demande des points pour un animal */

if yesanswer
Do you want to compare the evolution
of an animal with the reference sample?")
   then
     BEGIN;
      if yesanswer("Enter the data from a table ?")
         then
           begin;
             t = gettable("Name of the table to use: ");
             DO while not tableexists(t);
                type "ERROR this table do not exist !!!";
                t = gettable("Name of the table to use: ");
             end;
             add curve to evopsexegraph from col 0 of table(t)
                       VS col 1 of table(t);
           end;
         else
           begin;
             make table ANIMAL from col 0 of calculstat;
             delete row 1,2,3 of ANIMAL;
             i = 1;
             CONTINUER = true;
             DO while CONTINUER ;

               set AGE = GETNUMBER("Age (days) : ",TRUE);
               set VALEUR = GETNUMBER("Weight : ",TRUE);
               if (AGE = empty) or (VALEUR = empty)
                  then
                     CONTINUER = false;
                  else
                    begin;
                      insert row i of ANIMAL;
                      set row i col 0 of ANIMAL to AGE;
                      set row i col 1 of ANIMAL to VALEUR;
                      i = i + 1;
                    end;
             END;
             add curve to evopsexegraph from col 0 of ANIMAL
                       VS col 1 of ANIMAL;
             if yesanswer("Do you want to save this table?")
```

```
                then
                   begin;
                     t = gettable("Name of the table : ",True);
                     make table(t) from ANIMAL;
                   end;
          end;
      if reponse = "R"
        then
         set label of curve 5 of evopsexegraph to "Compared animal curve";
        else
          if reponse = "F" or reponse = "M"
            then
              set label of curve 7 of
                           evopsexegraph to "Compared animal curve";
            else
              if reponse = "T"
                then
                  set label of curve 9 of
                           evopsexegraph to "Compared animal curve";
      display evopsexegraph bottomkey height 0.9 width 0.65 at (0,0);
    END;

Estimation de Y à partir de X */

if yesanswer("Do you want an estimate of Y value for a given X ?")
   then
     begin;
       if yesanswer("For Male ?")
          then
            begin;
              if (reponse = "F") or (reponse = "T")
                 then
                   numcourbe = 6;
                 else
                   numcourbe = 4;
              do while true;
                y = $evalcurve("evopsexegraph",numcourbe);
                type nocr "y=", y;
                if yesanswer("Do you want to exit ?")
                   then doexit;
              end;
            end;
       if yesanswer("For Female ?")
          then
            begin;
              do while true;
                y = $evalcurve("evopsexegraph",3);
                type nocr "y=", y;
                if yesanswer("Do you want to exit ?")
                   then doexit;
              end;
            end;
     end;


if yesanswer("Do you want to save this graph ?")
   then
      begin;
        g = getgraph("Name under which the graph will be saved ?",True);
        make graph(g) from evopsexegraph;
```

```
      end;

deletion */

delete evopsexegraph;
delete evopsexe_init;
if tableexists("tabresum")
   then
      delete table tabresum ;
if tableexists("interfit")
   then
      delete interfit ;
if tableexists("tabresum2")
   then
      delete table tabresum2 ;
if tableexists("interfit2")
   then
      delete interfit2 ;
if tableexists("ANIMAL")
   then
      delete ANIMAL ;

d;
```

A N N E X E 7

F O R M A T P H Y S I Q U E D E S F I C H I E R S

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: ANIMAL

of File: ANIMAL

Organization: INDEXED

Type: RMS

Open: ANIMAL

Record Format: Fixed

Supersede: No

Record Size: 51 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
.NR_ANIMAL	CHARACTER	8		0
.CODE_ETUDE	CHARACTER	8		8
NUM_FOURNISSEUR	INTEGER SIGNED	4		16
SEXE	CHARACTER	1		20
ESPECE	CHARACTER	10		21
RACE	CHARACTER	20		31

Index Contents --

** NUM_ANIMAL is a 16 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

** CODE_ETUDE is a 8 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
CODE_ETUDE	CHARACTER	8

* NUM_FOURNISSEUR is a 4 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
NUM_FOURNISSEUR	INTEGER SIGNED	4

* ESPECE is a 10 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
ESPECE	CHARACTER	10

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: CL_CODE

of File: CL_CODE

Organization: INDEXED

Type: RMS

Open: CL_CODE

Record Format: Fixed

Supersede: No

Record Size: 16 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
CODE_BIDON	CHARACTER	8		0
CODE_DEF	CHARACTER	8		8

- Index Contents --

** CODE_BIDON is a 8 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
CODE_BIDON	CHARACTER	8

** CODE_DEF is a 8 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
CODE_DEF	CHARACTER	8

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: CL_NUMERO
of File: CL_NUMERO
Organization: INDEXED
Type: RMS
Open: CL_NUMERO
Record Format: Fixed
Supersede: No
Record Size: 16 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NR_LOT	CHARACTER	8		0
NR_DEF	CHARACTER	8		8

Index Contents --

* NR_LOT is a 8 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NR_LOT	CHARACTER	8

* NR_DEF is a 8 byte UNIQUE ALTERNATE ASCENDING index **

Segment	Type	Size
NR_DEF	CHARACTER	8

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: CONDEXPERIMENT

of File: CONDEXPERIMENT

Organization: INDEXED

Type: RMS

Open: CONDEXPERIMENT

Record Format: Fixed

Supersede: No

Record Size: 112 Bytes

-- Record Contents --

Item	Type	Size	Occ	Offset
NUM_COND_RES	INTEGER SIGNED	8		0
VOIE_INJECTION	CHARACTER	20		8
VOL_ADMINISTRE	FLOAT	8		28
TYPE_ECHANTILLON	CHARACTER	20		36
METHODE_ANALYSE	CHARACTER	25		56
VEHICULE_UTILISE	CHARACTER	25		81
ETAT_ALIMENTAIRE	CHARACTER	6		106

-- Index Contents --

** NUM_COND_RES is a 8 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NUM_COND_RES	INTEGER SIGNED	8

** TYPE_ECHANTILLON is a 20 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
TYPE_ECHANTILLON	CHARACTER	20

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: DETERMINANT
of File: DETERMINANT
Organization: INDEXED
Type: RMS
Open: DETERMINANT
Record Format: Fixed
Supersede: No
Record Size: 50 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
NOM_PARAM_BIO	CHARACTER	25		0
NOM_DET	CHARACTER	25		25

- Index Contents --

** NOM_PARAM_BIO is a 25 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
NOM_PARAM_BIO	CHARACTER	25

** NOM_DET is a 25 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
NOM_DET	CHARACTER	25

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: ELEMCOMMUNS

of File: ELEMCOMMUNS

Organization: SEQUENTIAL

Type: RMS

Open: ELEMCOMMUNS

Record Format: Fixed

Supersede: No

Record Size: 141 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
CODE_ETUDE	CHARACTER	8		0
DATE_DEBUT_ETUDE	INTEGER SIGNED	4		8
TYPE_ALIMENT	CHARACTER	20		12
RACE	CHARACTER	20		32
ESPECE	CHARACTER	10		52
ETAT	CHARACTER	8		62
VOIE_INJECTION	CHARACTER	20		70
TYPE_ECHANTILLON	CHARACTER	20		90
ETAT_ALIMENTAIRE	CHARACTER	6		110
VEHICULE_UTILISE	CHARACTER	25		116

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: ELEMHYPERVARIABLES

of File: ELEMHYPERVAR

Organization: INDEXED

Type: RMS

Open: ELEMHYPERVAR

Record Format: Fixed

Supersede: No

Record Size: 187 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
METHODE_ANALYSE	CHARACTER	25		16
JOUR_ETUDE	INTEGER SIGNED	4		41
DATE_RES	INTEGER SIGNED	4		45
.DATE_RES_CHAR	CHARACTER	4		45
VALEUR_RES	FLOAT	8		49
NOM_PARAM_BIO	CHARACTER	25		57
NOM_DET	CHARACTER	25		82
COMMENTAIRE	CHARACTER	80		107

- Index Contents --

** NUM_ANIMAL is a 16 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: ELEMVARIABLES

of File: ELEMVAR

Organization: INDEXED

Type: RMS

Open: ELEMVAR

Record Format: Fixed

Supersede: No

Record Size: 83 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
NR_ANIMAL	CHARACTER	8		16
SEXE	CHARACTER	1		24
DUREE	INTEGER SIGNED	2		25
AGE_ANIMAL	INTEGER SIGNED	2		27
PER_ACCLIMATATION	INTEGER SIGNED	2		29
DATE_ENTREE	INTEGER SIGNED	4		31
NOM_FOURNISSEUR	CHARACTER	20		35
NUM_FOURNISSEUR	INTEGER SIGNED	4		55
VOL_ADMINISTRE_CHAR	CHARACTER	8		59
.VOL_ADMINISTRE	FLOAT	8		59
CODE_ETUDE_BIDON	CHARACTER	8		67
NR_LOT	CHARACTER	8		75

Index Contents --

* NUM_ANIMAL is a 16 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	ESPECE
of File:	ESPECE
Organization:	INDEXED
Type:	RMS
Open:	ESPECE
Record Format:	Fixed
Supersede:	No
Record Size:	10 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
ESPECE	CHARACTER	10		0

Index Contents --

** ESPECE is a 10 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
ESPECE	CHARACTER	10

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: ETUDES
of File: ETUDES
Organization: INDEXED
Type: RMS
Open: ETUDES
Record Format: Fixed
Supersede: No
Record Size: 12 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
CODE_ETUDE	CHARACTER	8		0
DATE_DEBUT_ETUDE	INTEGER SIGNED	4		8

- Index Contents --

** CODE_ETUDE is a 8 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
CODE_ETUDE	CHARACTER	8

** DATE_DEBUT_ETUDE is a 4 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
DATE_DEBUT_ETUDE	INTEGER SIGNED	4

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: EVOPOIDSAGE

of File: EVOPOIDSAGE

Organization: SEQUENTIAL

Type: RMS

Open: EVOPOIDSAGE

Record Format: Fixed

Supersede: No

Record Size: 31 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
ESPECE	CHARACTER	10		0
RACE	CHARACTER	20		10
SEXE	CHARACTER	1		30

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	EVOPOIDSETUDE
of File:	EVOPOIDSETUDE
Organization:	SEQUENTIAL
Type:	RMS
Open:	EVOPOIDSETUDE
Record Format:	Fixed
Supersede:	No
Record Size:	8 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
CODE_ETUDE	CHARACTER	8		0

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: FICH_CONVERTI

of File: FICH_CONVERTI

Organization: SEQUENTIAL

Type: RMS

Open: FICH_CONVERTI

Record Format: Fixed

Supersede: No

Record Size: 71 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NOM_DET	CHARACTER	25		0
NR_ANIMAL	CHARACTER	8		25
SEXE	CHARACTER	1		33
JOUR_ETUDE	INTEGER SIGNED	4		34
NOM_PARAM_BIO	CHARACTER	25		38
VALEUR_RES	FLOAT	8		63

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: FOURNISSEUR

of File: FOURNISSEUR

Organization: INDEXED

Type: RMS

Open: FOURNISSEUR

Record Format: Fixed

Supersede: No

Record Size: 24 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
NUM_FOURNISSEUR	INTEGER SIGNED	4		0
NOM_FOURNISSEUR	CHARACTER	20		4

- Index Contents --

** NUM_FOURNISSEUR is a 4 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NUM_FOURNISSEUR	INTEGER SIGNED	4

** NOM_FOURNISSEUR is a 20 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
NOM_FOURNISSEUR	CHARACTER	20

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: FRUIT_PRE_SELECTION_5
of File: FRUIT_PRE_SELECTION_5
Organization: INDEXED
Type: RMS
Open: FRUIT_PRE_SELECTION_5
Record Format: Fixed
Supersede: No
Record Size: 8 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
NUM_COND_RES	INTEGER SIGNED	8		0

- Index Contents --

** NUM_COND_RES is a 8 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
NUM_COND_RES	INTEGER SIGNED	8

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: FRUIT_SELECTION_1

of File: FRUIT_SELECTION_1

Organization: INDEXED

Type: RMS

Open: FRUIT_SELECTION_1

Record Format: Fixed

Supersede: No

Record Size: 51 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
.NR_ANIMAL	CHARACTER	8		0
.CODE_ETUDE	CHARACTER	8		8
NUM_FOURNISSEUR	INTEGER SIGNED	4		16
ESPECE	CHARACTER	10		20
RACE	CHARACTER	20		30
SEXE	CHARACTER	1		50

- Index Contents --

** NUM_FOURNISSEUR is a 4 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
NUM_FOURNISSEUR	INTEGER SIGNED	4

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	FRUIT_SELECTION_2
of File:	FRUIT_SELECTION_2
Organization:	INDEXED
Type:	RMS
Open:	FRUIT_SELECTION_2
Record Format:	Fixed
Supersede:	No
Record Size:	20 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
NUM_FOURNISSEUR	INTEGER SIGNED	4		16

- Index Contents --

** NUM_FOURNISSEUR is a 4 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
NUM_FOURNISSEUR	INTEGER SIGNED	4

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	FRUIT_SELECTION_3
of File:	FRUIT_SELECTION_3
Organization:	SEQUENTIAL
Type:	RMS
Open:	FRUIT_SELECTION_3
Record Format:	Fixed
Supersede:	No
Record Size:	16 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	FRUIT_SELECTION_4
of File:	FRUIT_SELECTION_4
Organization:	INDEXED
Type:	RMS
Open:	FRUIT_SELECTION_4
Record Format:	Fixed
Supersede:	No
Record Size:	44 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
NUM_COND_RES	INTEGER SIGNED	8		16
DATE_RES	INTEGER SIGNED	4		24
VALEUR_RES	FLOAT	8		28
ETAT	CHARACTER	8		36

- Index Contents --

** NUM_ANIMAL is a 16 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

** NUM_COND_RES is a 8 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size

NUM_COND_RES	INTEGER SIGNED	8

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	FRUIT_SELECTION_5
of File:	FRUIT_SELECTION_5
Organization:	INDEXED
Type:	RMS
Open:	FRUIT_SELECTION_5
Record Format:	Fixed
Supersede:	No
Record Size:	44 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
DATE_RES	INTEGER SIGNED	4		16
VALEUR_RES	FLOAT	8		20
ETAT	CHARACTER	8		28
NUM_COND_RES	INTEGER SIGNED	8		36

-- Index Contents --

** NUM_ANIMAL is a 16 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: FRUIT_SELECTION_6

of File: FRUIT_SELECTION_6

Organization: SEQUENTIAL

Type: RMS

Open: FRUIT_SELECTION_6

Record Format: Fixed

Supersede: No

Record Size: 44 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
VALEUR_RES	FLOAT	8		0
ETAT	CHARACTER	8		8
NUM_ANIMAL	CHARACTER	16		16
NUM_COND_RES	INTEGER SIGNED	8		32
DATE_RES	INTEGER SIGNED	4		40

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: INT_COND_EXP

of File: INT_COND_EXP

Organization: INDEXED

Type: RMS

Open: INT_COND_EXP

Record Format: Fixed

Supersede: No

Record Size: 49 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
METHODE_ANIMAL	CHARACTER	41		0
.METHODE_ANALYSE	CHARACTER	25		0
.NUM_ANIMAL	CHARACTER	16		25
NUM_COND_RES	INTEGER SIGNED	8		41
.NUM_COND_RES_CHAR	CHARACTER	8		41

Index Contents --

* METHODE_ANIMAL is a 41 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
METHODE_ANIMAL	CHARACTER	41

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: INT_COND_EXP_TEMP

of File: INT_COND_EXP_TEMP

Organization: INDEXED

Type: RMS

Open: INT_COND_EXP_TEMP

Record Format: Fixed

Supersede: No

Record Size: 49 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
METHODE_ANIMAL	CHARACTER	41		0
.METHODE_ANALYSE	CHARACTER	25		0
.NUM_ANIMAL	CHARACTER	16		25
NUM_COND_RES	INTEGER SIGNED	8		41
.NUM_COND_RES_CHAR	CHARACTER	8		41

Index Contents --

* METHODE_ANIMAL is a 41 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
METHODE_ANIMAL	CHARACTER	41

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: NUM_MAX

of File: NUM_MAX

Organization: INDEXED

Type: RMS

Open: NUM_MAX

Record Format: Fixed

Supersede: No

Record Size: 12 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NUM_COND_RES	INTEGER SIGNED	8		0
NUM_FOURNISSEUR	INTEGER SIGNED	4		8

Index Contents --

* NUM_COND_RES is a 8 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NUM_COND_RES	INTEGER SIGNED	8

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: PARAMBIOL
of File: PARAMBIOL
Organization: INDEXED
Type: RMS
Open: PARAMBIOL
Record Format: Fixed
Supersede: No
Record Size: 33 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NOM_PARAM_BIO	CHARACTER	25		0
UNITE_MESURE	CHARACTER	8		25

Index Contents --

* NOM_PARAM_BIO is a 25 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NOM_PARAM_BIO	CHARACTER	25

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: POIDS

of File: POIDS

Organization: INDEXED

Type: RMS

Open: POIDS

Record Format: Fixed

Supersede: No

Record Size: 24 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
DATE_MESURE	INTEGER SIGNED	4		16
VALEUR_POIDS	INTEGER SIGNED	4		20

Index Contents --

** NUM_ANIMAL is a 16 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: POIDSINTER

of File: POIDSINTER

Organization: INDEXED

Type: RMS

Open: POIDSINTER

Record Format: Fixed

Supersede: No

Record Size: 24 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
DATE_MESURE	INTEGER SIGNED	4		16
VALEUR_POIDS	INTEGER SIGNED	4		20

Index Contents --

** NUM_ANIMAL is a 16 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: POSTTEST

of File: POSTTEST

Organization: INDEXED

Type: RMS

Open: POSTTEST

Record Format: Fixed

Supersede: No

Record Size: 20 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
POIDS_ORGANE	INTEGER SIGNED	4		16

Index Contents --

* NUM_ANIMAL is a 16 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: PRETEST
of File: PRETEST
Organization: INDEXED
Type: RMS
Open: PRETEST
Record Format: Fixed
Supersede: No
Record Size: 26 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
DATE_ENTREE	INTEGER SIGNED	4		16
DUREE	INTEGER SIGNED	2		20
AGE_ANIMAL	INTEGER SIGNED	2		22
PER_ACCLIMATATION	INTEGER SIGNED	2		24

- Index Contents --

** NUM_ANIMAL is a 16 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

** DATE_ENTREE is a 4 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size

DATE_ENTREE	INTEGER SIGNED	4

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: PRE_INT_COND_EXP

of File: PRE_INT_COND_EXP

Organization: INDEXED

Type: RMS

Open: PRE_INT_COND_EXP

Record Format: Fixed

Supersede: No

Record Size: 41 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NUM_COND_RES	INTEGER SIGNED	8		0
METHODE_ANALYSE	CHARACTER	25		8
VOL_ADMINISTRE_CHAR	CHARACTER	8		33
.VOL_ADMINISTRE	FLOAT	8		33

Index Contents --

** VOL_ADMINISTRE_CHAR is a 8 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
VOL_ADMINISTRE_CHAR	CHARACTER	8

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: RACE

of File: RACE

Organization: INDEXED

Type: RMS

Open: RACE

Record Format: Fixed

Supersede: No

Record Size: 20 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
RACE	CHARACTER	20		0

Index Contents --

** RACE is a 20 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
RACE	CHARACTER	20

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	REQUETE_BIOL
of File:	REQUETE_BIOL
Organization:	SEQUENTIAL
Type:	RMS
Open:	REQUETE_BIOL
Record Format:	Fixed
Supersede:	No
Record Size:	158 Bytes

- Record Contents --

Item	Type	Size	Occ	Offset
ESPECE	CHARACTER	10		0
RACE	CHARACTER	20		10
NOM_FOURNISSEUR	CHARACTER	20		30
SEXE	CHARACTER	1		50
AGE_ANIMAL	INTEGER SIGNED	2		51
DATE_ENTREE	INTEGER SIGNED	4		53
ETAT_ALIMENTAIRE	CHARACTER	6		57
TYPE_ECHANTILLON	CHARACTER	20		63
NOM_PARAM_BIO	CHARACTER	25		83
METHODE_ANALYSE	CHARACTER	25		108
OPERATEUR_DATE	CHARACTER	1		133
OPERATEUR_AGE	CHARACTER	1		134
TYPESTAT	CHARACTER	15		135
UNITE_MESURE	CHARACTER	8		150

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: RESULTAT

of File: RESULTAT

Organization: INDEXED

Type: RMS

Open: RESULTAT

Record Format: Fixed

Supersede: No

Record Size: 149 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
CLE_TGV	CHARACTER	53		0
.CLE_SELECTION	CHARACTER	41		0
..NUM_ANIMAL	CHARACTER	16		0
..NOM_PARAM_BIO	CHARACTER	25		16
.NUM_COND_RES_CHAR	CHARACTER	8		41
..NUM_COND_RES	INTEGER SIGNED	8		41
.DATE_RES_CHAR	CHARACTER	4		49
..DATE_RES	INTEGER SIGNED	4		49
VALEUR_RES	FLOAT	8		53
ETAT	CHARACTER	8		61
COMMENTAIRE	CHARACTER	80		69

Index Contents --

** CLE_TGV is a 53 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
CLE_TGV	CHARACTER	53

** CLE_SELECTION is a 41 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
CLE_SELECTION	CHARACTER	41

** NUM_ANIMAL is a 16 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

NUM_COND_RES_CHAR is a 8 byte REPEATING ALTERNATE ASCENDING index **

Segment	Type	Size
NUM_COND_RES_CHAR	CHARACTER	8

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: SELECTION_DETERMINANT
of File: SELECTION_DETERMINANT
Organization: INDEXED
Type: RMS
Open: SELECTION_DETERMINANT
Record Format: Fixed
Supersede: No
Record Size: 25 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NOM_PARAM_BIO	CHARACTER	25		0

Index Contents --

* NOM_PARAM_BIO is a 25 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NOM_PARAM_BIO	CHARACTER	25

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	STOCKAGE
of File:	STOCKAGE
Organization:	SEQUENTIAL
Type:	RMS
Open:	STOCKAGE
Record Format:	Fixed
Supersede:	No
Record Size:	15 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
TYPESTAT	CHARACTER	15		0

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: TABLE_MESURE
of File: TABLE_MESURE
Organization: INDEXED
Type: RMS
Open: TABLE_MESURE
Record Format: Fixed
Supersede: No
Record Size: 18 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
ENTREE	CHARACTER	5		0
FACTEUR	FLOAT	8		5
SORTIE	CHARACTER	5		13

Index Contents --

** ENTREE is a 5 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
ENTREE	CHARACTER	5

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: TAB_CHANGEMENTS

of File: TAB_CHANGEMENTS

Organization: INDEXED

Type: RMS

Open: TAB_CHANGEMENTS

Record Format: Fixed

Supersede: No

Record Size: 32 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
CLE_TAB	CHARACTER	16		0
.CODE_BIDON	CHARACTER	8		0
.CODE_DEF	CHARACTER	8		8
NR_LOT	CHARACTER	8		16
NR_DEF	CHARACTER	8		24

Index Contents --

** CLE_TAB is a 16 byte REPEATING PRIMARY ASCENDING index **

Segment	Type	Size
CLE_TAB	CHARACTER	16

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: TEST

of File: TEST

Organization: INDEXED

Type: RMS

Open: TEST

Record Format: Fixed

Supersede: No

Record Size: 36 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
NUM_ANIMAL	CHARACTER	16		0
TYPE_ALIMENT	CHARACTER	20		16

Index Contents --

* NUM_ANIMAL is a 16 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
NUM_ANIMAL	CHARACTER	16

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	TYPE_ALIMENT
of File:	TYPE_ALIMENT
Organization:	INDEXED
Type:	RMS
Open:	TYPE_ALIMENT
Record Format:	Fixed
Supersede:	No
Record Size:	20 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
TYPE_ALIMENT	CHARACTER	20		0

Index Contents --

** TYPE_ALIMENT is a 20 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
TYPE_ALIMENT	CHARACTER	20

R E C O R D R E P O R T

For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record:	TYPE_ECHANTILLON
of File:	TYPE_ECHANTILLON
Organization:	INDEXED
Type:	RMS
Open:	TYPE_ECHANTILLON
Record Format:	Fixed
Supersede:	No
Record Size:	20 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
TYPE_ECHANTILLON	CHARACTER	20		0

Index Contents --

TYPE_ECHANTILLON is a 20 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
TYPE_ECHANTILLON	CHARACTER	20

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: VEHICULE_UTILISE
of File: VEHICULE_UTILISE
Organization: INDEXED
Type: RMS
Open: VEHICULE_UTILISE
Record Format: Fixed
Supersede: No
Record Size: 25 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
VEHICULE_UTILISE	CHARACTER	25		0

Index Contents --

VEHICULE_UTILISE is a 25 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
VEHICULE_UTILISE	CHARACTER	25

R E C O R D R E P O R T
For DICTIONARY: 1DJA4:[VMMARC.BD]DIC_BASEDONNEE.PHD;

Record: VOIE_INJECTION
of File: VOIE_INJECTION
Organization: INDEXED
Type: RMS
Open: VOIE_INJECTION
Record Format: Fixed
Supersede: No
Record Size: 20 Bytes

Record Contents --

Item	Type	Size	Occ	Offset
VOIE_INJECTION	CHARACTER	20		0

Index Contents --

** VOIE_INJECTION is a 20 byte UNIQUE PRIMARY ASCENDING index **

Segment	Type	Size
VOIE_INJECTION	CHARACTER	20